Building Security In Maturity Model

BSIMM-V

Octubre 2013

Release 5.0

Autores

BSIMM4 y BSIMM-V: Gary McGraw, Ph.D., Sammy Migues y Jacob West
BSIMM1 a 3: Gary McGraw, Ph.D., Brian Chess, Ph.D. y Sammy Migues

Agradecimientos

Agradecemos a los ejecutivos de las 67 iniciativas de seguridad del software a nivel mundial que estudiamos para crear BSIMM-V. Estas incluyen a Adobe (Brad Arkin), Aetna (Jim Routh), Bank of America (Jim Apple), Box, Capital One (Keith Gordon), Comerica Bank (George Smirnoff), EMC (Eric Baize), Epsilon (Chris Ray), F-Secure (Antti Vähä-Sipilä), Fannie Mae (Stan Wisseman), Fidelity (David Smith), Goldman Sachs (Phil Venables), HSBC (Malcolm Kelly y Simon Hales), Intel, Intuit, JPMorgan Chase & Co. (Jeff Cohen), Lender Processing Services Inc., Marks and Spencer (Noel Dunne), Mashery (Chris Lippi), McAfee (James Ransome), McKesson (Mike Wilson), Microsoft (Steve Lipner), NetSuite (Brian Chess), Neustar (Jonathan Coombes), Nokia (Janne Uusilehto), Nokia Siemens Networks (Konstantin Shemyak), PayPal (Erick Lee), Pearson Learning Technologies (Aaron Weaver), QUALCOMM (Alex Gantman), Rackspace (Jim Freeman), Salesforce (Robert Fly), Sallie Mae (Jerry Archer), SAP (Gerhard Oswald), Sony Mobile (Per-Olof Persson), Standard Life (Alan Stevens), SWIFT (Peter De Gersem y Alain Desausoi), Symantec (Gary Phillips), Telecom Italia (Marco Bavazzano), Thomson Reuters (Timothy Mathias), TomTom (Xander Heemskerd), Vanguard (Samuel M. D'Amore, Jr.), Visa (Gary Warzala), VMware (Iain Mulholland), Wells Fargo (Steve Adegbite), y Zynga. Para aquellos que no podemos nombrar, ustedes saben quiénes son, y que sin los cuales no hubiéramos podido hacer lo que hicimos.

Agradecemos a Gabriele Giuseppini, David Harper, John Holland, Paco Hope, Matias Madou y Florence Mottay, quienes nos ayudaron con la recolección de datos en Europa. También a Andres Cools, Partha Dutta, Nabil Hannan, Jason Hills, Girish Janardhanudu, Troy Jones, Drew Kilbourne, Todd Lukens, Brian Mizelle, Kabir Mulchandani, Jason Rouse, Joel Scambray, Jay Schuman, Carl Schwarcz, Rajiv Sinha, Mike Ware, Caroline Wong y Dave Wong por la ayuda prestada en la recolección de datos en EE. UU. Agradecemos a Matteo Meucci (Minded Security), a Markus Schumacher (Virtual Forge), y a Susana Romaniz y equipo (Universidad Tecnológica Nacional Facultad Regional Santa Fe, Argentina) e Iván Arce (Fundación Dr. Manuel Sadosky, Argentina) por las traducciones al idioma italiano, alemán y castellano respectivamente. Agradecemos a Betsy Nichols (PlexLogic) por el análisis estadístico en BSIMM2.

Agradecemos a Pravir Chandra quien construyó un borrador de modelo de madurez bajo contrato de Fortify Software, dando así inicio a este proyecto. Agradecemos a John Steven por construir el primer marco de referencia para la seguridad del software, descripto en el capítulo 10 de *Software Security*. Agradecemos a John Steven, Roger Thornton, Mike Ware, Jim DelGrosso y Robert Hines por ayudarnos a elaborar el MRSS que aquí se describe.

Los datos para The Building Security In Maturity Model fueron recolectados por Cigital y HP Fortify. Traducciones del modelo BSIMM a cargo de VirtualForge (alemán), Minded Security (italiano), y Universidad Tecnológica Nacional Facultad Regional Santa Fe y Fundación Dr. Manuel Sadosky (castellano).

Licenciamiento de BSIMM-V

Resumen ejecutivo

Building Security In Maturity Model (BSIMM) es el resultado de un estudio de iniciativas de seguridad del software del mundo real, desarrollado a lo largo de varios años. Presentamos el modelo que se construyó a partir de los datos obtenidos en 67 iniciativas de seguridad del software, de empresas que comprenden, entre otras, a Adobe, Aetna, Bank of America, Box, Capital One, Comerica Bank, EMC, Epsilon, F-Secure, Fannie Mae, Fidelity, Goldman Sachs, HSBC, Intel, Intuit, JPMorgan Chase & Co., Lender Processing Services Inc., Marks and Spencer, Mashery, McAfee, McKesson, Microsoft, NetSuite, Neustar, Nokia, Nokia Siemens Networks, PayPal, Pearson Learning Technologies, QUALCOMM, Rackspace, Salesforce, Sallie Mae, SAP, Sony Mobile, Standard Life, SWIFT, Symantec, Telecom Italia, Thomson Reuters, TomTom, Vanguard, Visa, VMware, Wells Fargo y Zynga.

BSIMM es una vara para medir la seguridad del software. La mejor manera de utilizar BSIMM es emplearlo para comparar y contrastar su propia iniciativa con los datos de lo que otras organizaciones están haciendo en las áreas que conciernen al modelo. Así podrá identificar sus propias metas y objetivos, y recurrir luego a BSIMM para determinar qué actividades adicionales tienen más sentido para usted.

Los datos de BSIMM muestran que las iniciativas de mayor madurez están bien equilibradas (es decir, llevan a cabo numerosas actividades de las doce prácticas descritas por el modelo). El modelo también describe cómo evolucionan, cambian y mejoran a lo largo del tiempo las iniciativas maduras de seguridad del software.

BSIMM-V

Contenido

*B*uilding Security In Maturity Model (BSIMM, pronunciación *bisimm*) es un estudio de iniciativas existentes de seguridad del software. Al cuantificar las prácticas de muchas organizaciones diferentes, se puede describir la base común compartida por varias de ellas, así como la variación que hace que cada una sea única. Nuestro objetivo es ayudar a la vasta comunidad de seguridad del software a planificar, llevar a cabo y medir sus propias iniciativas. BSIMM no es una guía de los "cómo", ni una prescripción aplicable a todos. BSIMM es, en cambio, un reflejo de lo más avanzado en seguridad del software.

Comenzamos con una breve descripción de la función y la importancia de una iniciativa de seguridad del software. A continuación, explicamos nuestro modelo y el método que utilizamos para cuantificar el estado de una iniciativa. Desde el comienzo del estudio BSIMM en 2008, hemos estudiado 72 iniciativas, que comprenden 166 mediciones distintas, dado que algunas empresas utilizan BSIMM para medir cada una de sus unidades de negocio y otras se midieron más de una vez. Para asegurar la continuidad de la relevancia de los datos que reportamos, excluimos las mediciones con antigüedad superior a 48 meses de BSIMM-V. El conjunto de datos actuales comprende 161 mediciones diferentes recolectadas en 67 empresas. Gracias a la repetición de mediciones, reportamos no solo las prácticas actuales, sino también la manera en que algunas iniciativas evolucionaron durante un período de años. Dedicamos la última parte del documento a brindar una explicación detallada de las 112 actividades que actualmente componen nuestro modelo y un resumen de los datos en bruto que recolectamos. Revisamos la descripción de cada actividad de BSIMM-V y también añadimos una nueva actividad en el modelo.

Nuestro trabajo con el modelo BSIMM muestra que medir una iniciativa de seguridad del software de una empresa es posible y muy útil. Las mediciones BSIMM se pueden utilizar para planificar, estructurar y ejecutar la evolución de una iniciativa de seguridad del software. A lo largo del tiempo, las empresas participantes en el proyecto BSIMM muestran una mejora mensurable en esa área.

Introducción

La seguridad del software comenzó a prosperar como una disciplina independiente de la seguridad en las computadoras y en las redes a finales de la década de 1990. Los investigadores comenzaron a poner más énfasis en estudiar la manera en que un programador puede contribuir a mejorar la seguridad de un sistema informático o a, involuntariamente, socavarla. ¿Qué tipo de errores de programación (*bugs*) y defectos de diseño (*flaws*) conducen a los problemas de seguridad? ¿Cómo podemos identificar los problemas de manera sistemática?

A mediados de la década siguiente, comenzó a surgir el consenso de que la creación de software seguro requiere más que solo personas inteligentes esforzándose. Conseguir seguridad significa que esté involucrada en el proceso de desarrollo de software.

Desde entonces, los profesionales han descubierto que el proceso por sí solo tampoco es suficiente. La seguridad del software también abarca aspectos de negocio, sociales y de las organizaciones. Usamos la expresión "iniciativa de seguridad del software" para hacer referencia a todas las actividades llevadas a cabo con el propósito de construir software seguro.

BSIMM-V

El propósito de BSIMM es cuantificar las actividades llevadas a cabo por iniciativas reales de seguridad del software. Debido a que estas iniciativas hacen uso de diferentes metodologías y terminologías, BSIMM requiere un marco de referencia que nos permita describir todas las iniciativas de una manera uniforme. Nuestro Modelo de Referencia para la Seguridad del Software (MRSS) y sus descripciones de las actividades proporciona un vocabulario común para explicar los elementos más destacados de una iniciativa de seguridad del software, lo que nos permite comparar iniciativas que utilizan diferente terminología, operan a diferentes escalas, existen en diferentes mercados verticales o crean diferentes productos.

Clasificamos nuestro trabajo como un modelo de madurez ya que mejorar la seguridad del software casi siempre significa cambiar la forma en que trabaja una organización, algo que no sucede de la noche a la mañana. Entendemos que no todas las organizaciones necesitan alcanzar las mismas metas de seguridad, pero creemos que todas ellas se pueden beneficiar del uso de la misma vara para medir.

BSIMM-V es la quinta versión principal del modelo BSIMM. Incluye descripciones actualizadas de actividades, una nueva actividad, datos de 67 empresas, y un estudio longitudinal.

Audiencia

BSIMM está pensado para ser utilizado por cualquier persona responsable de la creación y ejecución de una iniciativa de seguridad del software. Observamos que las iniciativas de seguridad del software exitosas normalmente son llevadas a cabo por un ejecutivo prominente, que responde a los más altos niveles directivos de la organización. Estos ejecutivos lideran un grupo interno que llamamos Grupo de Seguridad del Software (GSS), encargado de ejecutar directamente o de facilitar las actividades descriptas en BSIMM. BSIMM está escrito pensando en el GSS y en su liderazgo.

Esperamos que los lectores estén familiarizados con la literatura de seguridad del software. Es posible familiarizarse con muchos conceptos mediante la lectura de Software Security y de The Security Development Lifecycle. BSIMM no intenta explicar los conceptos básicos de seguridad del software, describir su historia o proporcionar referencias a la literatura en constante expansión. Es poco probable que se pueda tener éxito con BSIMM si no se está familiarizado con esta literatura.

Método

Construimos la primera versión de BSIMM a mediados de 2008, del siguiente modo:

- Nos basamos en nuestro propio conocimiento de las prácticas de seguridad del software para crear el Modelo de Referencia para la Seguridad del Software. (Presentamos el modelo de referencia en la página 21).
- Realizamos una serie de nueve entrevistas presenciales con ejecutivos a cargo de las iniciativas de seguridad del software. A partir de estas entrevistas, identificamos un conjunto de actividades comunes, que organizamos con la guía del Modelo de Referencia para la Seguridad del Software.
- A continuación, creamos fichas de puntaje para cada una de las nueve iniciativas que muestran qué actividades lleva a cabo cada una de ellas. Con el fin de validar nuestro trabajo, solicitamos a cada participante la revisión del marco de referencia, las prácticas y la ficha de puntaje que creamos para su iniciativa.

En este punto del proyecto, utilizamos la misma técnica de entrevista para llevar a cabo evaluaciones BSIMM en 63 empresas adicionales (un total de 72 empresas). (En nueve casos, calculamos el puntaje de una empresa grande asignando puntajes a un número de las principales divisiones, y luego los combinamos en un puntaje único para la empresa). A partir de BSIMM-V, introducimos un requisito de frescura de datos para excluir mediciones anteriores a 48 meses. Este requisito hizo que cinco empresas fueran removidas de BSIMM-V (Aon, The Depository Trust & Clearing Corporation (DTCC), Google, Scripps Networks, y un participante anónimo), resultando en un conjunto de datos de 161 mediciones diferentes recolectadas en 67 empresas. A medida que los datos envejezcan, pretendemos reducir la ventana de frescura a 36 meses para su mejor alineamiento con los ciclos de negocio.

Utilizamos los puntajes resultantes para refinar el conjunto de actividades y su ubicación dentro del modelo de referencia. Además realizamos un segundo conjunto completo de entrevistas con 21 participantes, a fin de estudiar cómo habían cambiado sus iniciativas a lo largo del tiempo. Cuatro participantes han pasado por tres de estas mediciones.

También publicamos las observaciones acerca de subgrupos (tales como industrias verticales) cuando nuestro tamaño de la muestra para el subgrupo es lo suficientemente grande como para garantizar el anonimato de los participantes.

Mantuvimos la confidencialidad de las fichas de puntaje individuales de las empresas, pero publicamos los datos agregados que describen el número de veces que observamos cada actividad (ver página 67)

Al ser un modelo descriptivo, el objetivo de BSIMM es solo observar e informar. Nos gusta decir que caminamos por la selva para ver qué podíamos descubrir, y descubrimos que "los monos comen bananas en X de las Y selvas que visitamos". Observe que BSIMM no informa que "usted solo debería comer bananas amarillas", "no correr mientras se come una banana", "no robar las bananas de sus vecinos", ni emite cualquier otro juicio de valor. Observaciones simples, informadas simplemente.

Nuestro enfoque de "solo los hechos" no es nuevo en la ciencia ni en la ingeniería, pero no ha sido aplicado a esta escala en el dominio de la seguridad del software. Trabajos previos describen la experiencia de una única organización u ofrecen orientación normativa sobre la base de una combinación de experiencia y opinión personales.

Participantes

Las 67 organizaciones participantes proceden de doce industrias verticales (con cierta superposición): servicios financieros (26), proveedores independientes de software (25), servicios en la nube (16), empresas de tecnología (14), telecomunicaciones (5), venta al por menor (4), seguridad (4), salud (3), medios de comunicación (3), seguros (2), energía (1), y proveedores de servicios de Internet (1). De las 67 empresas, la lista de aquellas que, amablemente, aceptaron ser identificadas incluye: Adobe, Aetna, Bank of America, Box, Capital One, Comerica Bank, EMC, Epsilon, F-Secure, Fannie Mae, Fidelity, Goldman Sachs, HSBC, Intel, Intuit, JPMorgan Chase & Co., Lender Processing Services Inc., Marks and Spencer, Mashery, McAfee, McKesson, Microsoft, NetSuite, Neustar, Nokia, Nokia Siemens Networks, PayPal, Pearson Learning Technologies, QUALCOMM, Rackspace, Salesforce, Sallie Mae, SAP, Sony Mobile, Standard Life, SWIFT, Symantec, Telecom Italia, Thomson Reuters, TomTom, Vanguard, Visa, VMware, Wells Fargo y Zynga.

En promedio, los 67 participantes pusieron en práctica iniciativas de seguridad del software durante casi 4,25 años al momento de la medición (algunas de ellas están por estrenar una primera medición, mientras que la iniciativa más antigua cumple dieciocho años en octubre de 2013). Las 67 empresas concuerdan en que el éxito de sus programas depende de que se cuente con un grupo interno dedicado a la seguridad del software -el GSS. El tamaño promedio del GSS es de 14,78 personas (el menor es de 1; el mayor es de 100; la mediana es de 7) con un "satélite" (desarrolladores, arquitectos y personas de la organización que participan directamente y promueven la seguridad del software) de 29,6 personas (el menor es de 0; el mayor es de 400; la mediana es de 4). El número promedio de desarrolladores en las empresas participantes es de 4,190 personas (el menor es de 11; el mayor es de 30.000; la mediana es de 1.600). Esto supone un porcentaje promedio del GSS respecto de los desarrolladores de 1,4 %.

Teniendo en cuenta todo lo dicho, BSIMM describe el trabajo de 975 miembros de GSS trabajando con un satélite de 1.953 personas para asegurar el software desarrollado por 272.358 desarrolladores.

Objetivos

Creamos BSIMM con el fin de aprender cómo funcionan las iniciativas de seguridad del software y proporcionar un recurso para quienes buscan crear o mejorar su propia iniciativa.

En general, cualquier iniciativa de seguridad del software será creada pensando en algunas metas de alto nivel. BSIMM resulta apropiado si sus metas de negocio para la seguridad del software incluyen:

- Decisiones fundamentadas para la gestión de riesgo
- Claridad acerca de qué es "hacer lo correcto" para cada uno de los involucrados en la seguridad del software
- Reducción de costos a través de procesos estándares y repetibles
- Mejora de la calidad del código

Si cuenta con objetivos claramente expresados y realiza un seguimiento de las prácticas utilizando métricas adaptadas a su propia iniciativa de seguridad del software, puede emplear BSIMM como una herramienta de medición para guiarla.

Inculcar la seguridad del software dentro de una organización requiere de una planificación cuidadosa y siempre implica un cambio amplio en ella. Utilizando BSIMM como una guía para su propia iniciativa de seguridad del software, usted puede aprovechar los años de experiencia capturados en el modelo. Debería, sin embargo, adaptar las actividades que describe BSIMM a su propia organización (considerando cuidadosamente los objetivos que documentamos). Tenga en cuenta que ninguna organización lleva a cabo todas las actividades descriptas en BSIMM.

Terminología

La nomenclatura fue siempre un problema en la seguridad informática, y la seguridad del software no es una excepción. Hay una serie de términos que utilizamos en BSIMM que tienen un significado especial para nosotros. Aquí están algunos de los términos más importantes, que usamos a lo largo de todo el documento:
Actividad: Acciones llevadas a cabo o facilitadas por el GSS como parte de una práctica. En BSIMM, las actividades se dividen en tres niveles de madurez. Cada actividad está asociada directamente con un objetivo.
Dominio: Uno de los cuatro agrupamientos principales en el MRSS. Los dominios son cuatro: Gobernanza, Inteligencia, Puntos de Contacto con el SSDL (SSDL Touchpoints) y Despliegue. Consultar la sección "Modelo de Referencia para la Seguridad del Software".
Práctica: Una de las doce categorías de actividades BSIMM. Cada dominio del MRSS tiene tres prácticas; las actividades en cada práctica se dividen en tres niveles, que corresponden a la madurez. Consultar la sección "Modelo de Referencia para la Seguridad del Software".
Satélite: Un grupo interesado y comprometido de desarrolladores, arquitectos, administradores de software y testers que tienen una afinidad natural hacia la seguridad del software, y quienes están organizados y promovidos mediante una iniciativa de seguridad del software.
Ciclo de vida de desarrollo de software seguro (Secure Software Development Lifecycle, SSDL): Cualquier SDLC que integre puntos de control y actividades de seguridad del software.
Security Development Lifecycle (SDL): Término utilizado por Microsoft para describir su ciclo de vida de desarrollo de software seguro.
Modelo de Referencia para la Seguridad del Software (MRSS) (Software Security Framework, SSF): La estructura básica subyacente a BSIMM, compuesta por doce prácticas divididas en cuatro dominios. Consultar la sección "Modelo de Referencia para la Seguridad del Software".

Grupo de seguridad del software (GSS) (Software Security Group, GSS): El grupo interno encargado de llevar a cabo y facilitar la seguridad del software. Observamos que el primer paso de una iniciativa de seguridad del software es la formación de un GSS.
Iniciativa de seguridad del software: Programa aplicado a toda una organización, cuyo objetivo es inculcar, medir, gestionar y hacer evolucionar actividades de seguridad del software en forma coordinada. También conocido en la literatura como un "programa empresarial de seguridad del software" (consulte el capítulo 10 de *Software Security*).

Roles

Determinar quién se supone que debe llevar a cabo las actividades descriptas en BSIMM es una parte importante del trabajo a realizar en cualquier iniciativa de seguridad del software.

Liderazgo ejecutivo

Es de interés primordial identificar y potenciar a un ejecutivo prominente para que tenga a su cargo gestionar las operaciones, obtener los recursos y proporcionar la cobertura política a la iniciativa de seguridad del software. Las propuestas de seguridad del software promovidas y lideradas únicamente por los desarrolladores y sus gerentes directos tienen un pobre historial en el mundo real. Del mismo modo, las iniciativas encabezadas con recursos provenientes de un grupo ya existente de seguridad de redes, a menudo tienen serios problemas a la hora de interactuar con los grupos de desarrollo. Al identificar a un ejecutivo prominente y ponerlo directamente a cargo de la seguridad del software, usted aborda dos cuestiones básicas de gestión: responsabilidad y empoderamiento. Además, crea un lugar dentro de la organización donde la seguridad del software se puede arraigar y comenzar a prosperar.

Los ejecutivos a cargo de las iniciativas de seguridad del software que estudiamos tienen una variedad de títulos, entre ellos: director de Gestión de Riesgo y de Seguridad de las Tecnologías de la Información, oficial principal de Riesgo de la Información, director de Controles de Aplicación, vicepresidente de Productos y Operaciones, gerente de Seguridad de Productos, gerente senior de Seguridad de Productos, vicepresidente senior de Gestión de Riesgo Global, director global de Mercados de Seguridad de la Información, vicepresidente senior de Seguridad de la Información, y Oficial Principal de Seguridad de la Información (CISO). En las empresas estudiadas también observamos una dispersión bastante amplia respecto de dónde está situado exactamente el GSS. En particular, diecinueve GSS se encuentran bajo la dependencia del Oficial Principal de la Información (CIO), quince bajo la del Oficial Principal de Tecnología (CTO), catorce reportan al Oficial Principal de Operaciones (COO), cuatro existen en la dirección del Departamento Legales o la oficina de Cumplimiento y Gestión de Riesgo, tres reportan al Oficial Principal de Estrategia (CSO), y uno reporta directamente al Fundador o al Director Ejecutivo (CEO). Algunas de las compañías que estudiamos no especificaron dónde se ubica su GSS dentro la organización general.

Grupo de seguridad del software (GSS)

El rol que sigue en importancia al de ejecutivo a cargo, en una iniciativa de seguridad del software, es el del grupo de seguridad del software. Cada uno de los 67 programas que describimos en BSIMM tiene un Grupo de Seguridad del Software (GSS). La realización exitosa de las actividades de BSIMM, sin un GSS, es muy poco probable (y hasta la fecha jamás se observó un caso así), por lo que es necesario crear un GSS antes de comenzar con la adopción de las actividades de BSIMM. Los mejores miembros del GSS son personas de seguridad del software, a menudo, imposibles de encontrar. Si usted debe crear los perfiles de seguridad del software desde cero, comience con los desarrolladores y capacítelos en seguridad. No intente comenzar con personas provenientes de seguridad de redes, capacitándolos en el software, los compiladores, los SDLC, el seguimiento de errores de programación y todo lo relativo al mundo del software. Ningún nivel de experiencia en la seguridad tradicional puede superar la falta de conocimientos sobre software.

Los GSS se organizan en una variedad de formas y tamaños. Todos los buenos GSS incluyen tanto a personas con una profunda experiencia en programación como a personas con conocimientos en arquitecturas. Como se verá más adelante, la seguridad del software no puede enfocarse solo en la búsqueda de errores de programación específicos, tales como los que lista el *OWASP Top Ten*. La revisión de código es una buena práctica muy importante, y para llevarla a cabo usted debe entender el código en profundidad (y ni hablar de comprender la enorme cantidad de errores de programación que tienen impacto en la seguridad). Sin embargo, los mejores revisores de código a veces resultan ser muy pobres arquitectos de software, y pedirles que realicen un Análisis de Riesgo de la Arquitectura solo se traducirá en miradas en blanco. Asegúrese de cubrir en su GSS las capacidades relacionadas con la arquitectura así como lo hace con las de programación. Por último, muy a menudo se espera del GSS que guíe, entrene y trabaje directamente con cientos de desarrolladores. Es imprescindible que al menos una parte de los miembros del GSS cuente con habilidades de comunicación, capacidad de enseñanza y sentido común. Para obtener más información sobre este tema, véase nuestro artículo en InformIT *"You Really Need an GSS"* <http://bit.ly/7dqCn8>.

Si bien no hay dos de las 67 empresas examinadas que tuviesen exactamente la misma estructura de GSS (lo que hace pensar que no hay una manera única y determinada de estructurar un GSS), sí observamos algunos puntos en común que vale la pena mencionar. Desde una visión de alto nivel, se presentan tres variantes principales en la organización de un GSS: de acuerdo con las tareas técnicas del SDLC, de acuerdo con las tareas operativas, y de acuerdo con las unidades internas de negocio. Algunos GSS están altamente distribuidos a lo largo de toda la empresa, y otros están muy centralizados y orientados por políticas. Si analizamos todos los GSS de nuestro estudio, existen varios "subgrupos" comunes que se observan a menudo: personas dedicadas a las políticas, la estrategia y las métricas; grupos de "servicios" internos que (a menudo en forma separada) se encargan de las herramientas, las pruebas de penetración, y el desarrollo y guía del "middleware"; grupos de respuesta a incidentes; grupos responsables de desarrollar y brindar capacitación; grupos a cargo del marketing y la comunicación externa; y grupos de control de proveedores.

En las estadísticas presentadas anteriormente, observamos una relación promedio entre el número de integrantes del GSS y el de desarrolladores, muy cercana y por debajo al 1,4 % en el grupo completo de 67 empresas que estudiamos. Esto significa, promediando los porcentajes en cada participante, un miembro del GSS por cada 71 desarrolladores. La mayor proporción fue de un 27,27 % y la menor, de un 0,03 %. Considerando los datos en términos de casos reales, el tamaño promedio del GSS entre las 67 empresas fue de 14,7 personas (el menor fue de 1, el mayor fue de 100, y la mediana fue de 7).

Todos los demás

Los participantes en la encuesta detallaron quiénes más están involucrados en el ciclo de vida de desarrollo de software como una forma de abordar la seguridad del software:

- **Constructores** –incluidos desarrolladores, arquitectos y sus gerentes–, que deben poner en práctica la ingeniería de seguridad para garantizar que los sistemas que construyen puedan ser defendidos y no estén llenos de agujeros de seguridad. El GSS interactuará directamente con ellos cuando se lleven a cabo las actividades descriptas en BSIMM. En términos generales, a medida que una organización madura, el GSS intenta empoderar a los constructores para que puedan llevar a cabo la mayor parte de las actividades de BSIMM por sí mismos, con la ayuda del GSS en casos especiales y supervisándolos. En esta versión de BSIMM, a menudo no se señala explícitamente si una determinada actividad es llevada a cabo por el GSS, por los desarrolladores o por los *testers*, aunque en algunos casos intentamos aclarar, en las prácticas, las responsabilidades en las metas asociadas con los niveles de las actividades. Usted debería elaborar un enfoque que se adecúe a su organización y que tenga en cuenta la carga de trabajo y su ciclo de vida del software.

- ***Testers***, relacionados con las pruebas y la verificación de rutina, que deben hacer lo posible para poner su atención en los problemas de seguridad. Algunas de las actividades de BSIMM en la práctica *Pruebas de Seguridad* pueden ser llevadas a cabo directamente por el responsable de Aseguramiento de Calidad (QA).
- **Operadores**, que deben continuar diseñando redes adecuadas, protegerlas y mantenerlas operativas. Como veremos en el dominio *Despliegue* del MRSS, la seguridad del software no termina cuando el software se "despacha".
- **Administradores**, que deben comprender la naturaleza distribuida de los sistemas modernos y comenzar a poner en práctica el principio de "privilegio mínimo", especialmente cuando se trata de aplicaciones que ellos alojan o que se publican como servicios en la nube.
- **Ejecutivos y mandos medios**, incluidos los responsables de la línea de negocio y los gerentes de producto, que deben entender cómo la inversión temprana en el diseño y el análisis de la seguridad afectan el grado en el que los usuarios confiarán en sus productos. Los requisitos del negocio deberían abordar explícitamente las necesidades de la seguridad. En la actualidad, cualquier negocio importante depende del software para trabajar. La seguridad del software es una necesidad empresarial.
- **Proveedores**, incluidos los que suministran software comercial enlatado, software personalizado y *Software-as-a-Service* (SaaS), quienes están cada vez más sujetos a los acuerdos de nivel de servicio (SLA) y a revisiones (como vBSIMM) que ayudan a garantizar que los productos son el resultado de un SDLC seguro.

Los participantes tienen la sensación de que los constructores son las personas más importantes que hay que sumar a fin de lograr, en el corto plazo, avances en la seguridad del software. Sólo empujando la visión de la seguridad más allá de las operaciones habituales comenzaremos a construir sistemas de software que puedan resistir ataques.

Identificación del satélite

Además del GSS, muchos programas de seguridad del software identificaron a una serie de personas (a menudo desarrolladores, *testers* y arquitectos) que comparten un interés básico en la seguridad del software pero que no están directamente incluidos en el GSS. Llamamos a este grupo el *satélite*.

A veces el satélite se encuentra distribuido ampliamente, con uno o dos miembros de cada grupo de productos. Otras veces el satélite está más centralizado y se reúne regularmente para comparar notas, aprender nuevas tecnologías y ampliar la comprensión de la seguridad del software en una organización. Identificar y fomentar un satélite fuerte es importante para el éxito de muchas iniciativas de seguridad del software (pero no de todas). Algunas actividades de BSIMM están explícitamente orientadas al satélite.

Es de particular interés notar que las diez empresas con mayor puntaje BSIMM poseen un satélite (100 %), con un tamaño promedio de satélite de 148 personas. Además de las diez primeras, 29 de las 57 empresas restantes tienen un satélite (51 %). De las diez empresas con los puntajes BSIMM más bajos, sólo 2 tienen un satélite (20 %), de un tamaño promedio de menos de media persona. Esto sugiere que a medida que una iniciativa de seguridad del software madura, sus actividades se distribuyen e institucionalizan dentro de la estructura de la organización. Entre nuestra población de 67 empresas, las iniciativas comienzan, en un principio, con un satélite centralizado y especializado, y tienden a evolucionar hacia uno descentralizado y distribuido (con un GSS en el centro orquestando las cosas).

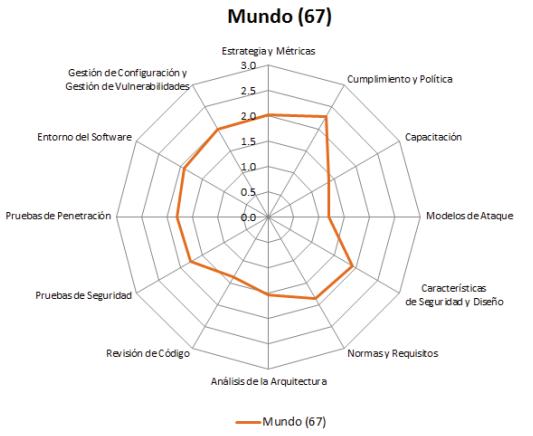
Uso de BSIMM-V

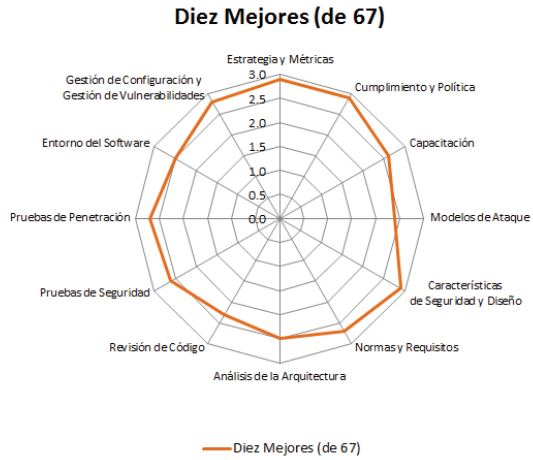
BSIMM describe 112 actividades que cualquier organización puede poner en práctica. Las actividades se describen en términos del MRSS, el cual identifica doce prácticas agrupadas en cuatro dominios. Cada actividad tiene asociado un objetivo.

Resultados de BSIMM-V

Los datos BSIMM arrojan resultados analíticos muy interesantes. Tenemos aquí dos gráficos radiales que muestran el nivel de madurez promedio, en cada una de las doce prácticas, de un cierto número de organizaciones. El primer gráfico muestra los datos de las 67 empresas en BSIMM-V. El segundo gráfico muestra los datos de las diez empresas principales según lo determinado por el puntaje "bruto" calculado por la suma de las actividades.

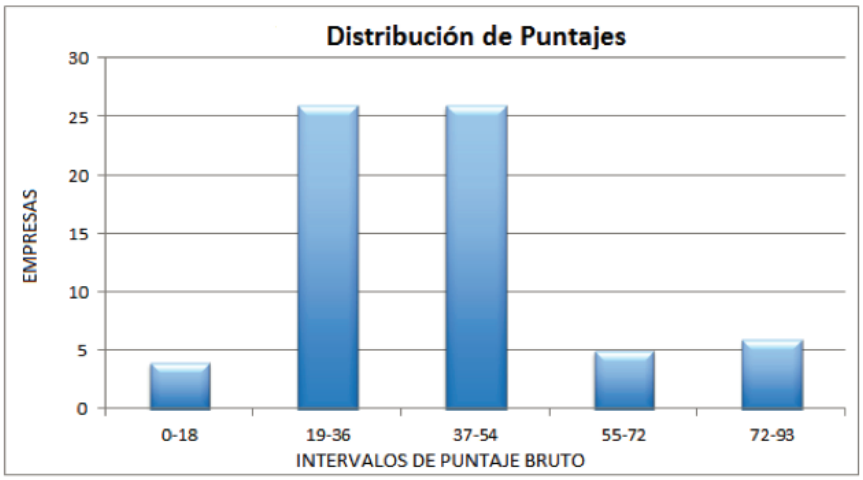
Los gráficos radiales se crearon destacando el mayor nivel de actividad dentro de una práctica para una empresa determinada (una "marca de agua máxima"), y luego promediando los puntajes de un grupo de empresas, lo que resulta en doce números (uno para cada práctica). El gráfico radial tiene doce ejes correspondientes a los doce prácticas. Tenga en cuenta que en todos estos diagramas, el nivel 3 (borde exterior) se considera más maduro que el nivel 0 (el punto central). Por supuesto que sería posible realizar otros análisis más sofisticados, por eso continuamos experimentando con ponderaciones según el nivel, la normalización por número de actividades y otros esquemas.





Al calcular un puntaje bruto para cada empresa dentro del estudio, también podemos observar la madurez relativa y la madurez promedio de una empresa respecto a las demás. Hasta la fecha, el rango de los puntajes observados es [13, 93].

El siguiente gráfico muestra la distribución de los puntajes entre la población de 67 empresas participantes. Para hacer este gráfico, dividimos los puntajes en cinco intervalos iguales. Como puede ver, los puntajes representan una curva en forma de campana un poco sesgada.



Nos complace que el estudio BSIMM continúe creciendo (el conjunto de datos creció algo más del 75 % desde la publicación de BSIMM4, y es más de 18 veces el tamaño que tenía en la publicación original). Tenga en cuenta que una vez que superamos un tamaño de muestra de 30 empresas, comenzamos a aplicar análisis estadístico para obtener resultados estadísticamente significativos.

La Ficha de Puntaje BSIMM-V que sigue muestra el número de veces que se observaron cada una de las 112 actividades dentro de los datos de BSIMM-V. En la página 67 encontrará una versión ampliada de este gráfico.

Gobernanza		Inteligencia		Puntos de Contacto con el SSDL		Despliegue	
Actividad	Observado	Actividad	Observado	Actividad	Observado	Actividad	Observado
[SM1.1]	44	[AM1.1]	21	[AA1.1]	56	[PT1.1]	62
[SM1.2]	34	[AM1.2]	43	[AA1.2]	35	[PT1.2]	51
[SM1.3]	34	[AM1.3]	30	[AA1.3]	24	[PT1.3]	43
[SM1.4]	57	[AM1.4]	12	[AA1.4]	42	[PT2.2]	24
[SM1.6]	36	[AM1.5]	42	[AA2.1]	10	[PT2.3]	27
[SM2.1]	26	[AM1.6]	16	[AA2.2]	8	[PT3.1]	13
[SM2.2]	31	[AM2.1]	7	[AA2.3]	20	[PT3.2]	8
[SM2.3]	27	[AM2.2]	11	[AA3.1]	11		
[SM2.5]	20	[AM3.1]	4	[AA3.2]	4		
[SM3.1]	16	[AM3.2]	6				
[SM3.2]	6						
[CP1.1]	43	[SFD1.1]	54	[CR1.1]	24	[SE1.1]	34
[CP1.2]	52	[SFD1.2]	53	[CR1.2]	34	[SE1.2]	61
[CP1.3]	45	[SFD2.1]	26	[CR1.4]	50	[SE2.2]	31
[CP2.1]	24	[SFD2.2]	29	[CR1.5]	23	[SE2.4]	25
[CP2.2]	28	[SFD3.1]	9	[CR1.6]	25	[SE3.2]	10
[CP2.3]	29	[SFD3.2]	13	[CR2.2]	10	[SE3.3]	9
[CP2.4]	25	[SFD3.3]	9	[CR2.5]	15		
[CP2.5]	35			[CR2.6]	18		
[CP3.1]	14			[CR3.2]	4		
[CP3.2]	11			[CR3.3]	6		
[CP3.3]	8			[CR3.4]	1		
[T1.1]	50	[SR1.1]	48	[ST1.1]	51	[CMVM1.1]	59
[T1.5]	29	[SR1.2]	43	[ST1.3]	55	[CMVM1.2]	59
[T1.6]	23	[SR1.3]	45	[ST2.1]	27	[CMVM2.1]	50
[T1.7]	33	[SR1.4]	27	[ST2.4]	13	[CMVM2.2]	44
[T2.5]	9	[SR2.2]	23	[ST3.1]	11	[CMVM2.3]	30
[T2.6]	13	[SR2.3]	19	[ST3.2]	8	[CMVM3.1]	6
[T2.7]	9	[SR2.4]	19	[ST3.3]	6	[CMVM3.2]	6
[T3.1]	4	[SR2.5]	22	[ST3.4]	5	[CMVM3.3]	2
[T3.2]	4	[SR3.1]	8	[ST3.5]	7		
[T3.3]	8	[SR3.2]	12				
[T3.4]	9						
[T3.5]	5						

Nuevas actividades para una observación BSIMM6

Por segunda vez en el proyecto BSIMM, observamos una actividad que el modelo aún no capturaba. Por lo tanto se añadió de aquí en más una nueva actividad en el modelo. Las observaciones relacionadas con esta actividad se informarán por primera vez en los datos publicados en BSIMM6, pero describimos la actividad en esta versión.

Durante las mediciones, se realizó un activo seguimiento de las dos actividades nuevas descritas por primera vez en BSIMM4; sin embargo, existe un cierto retraso en los datos respecto de estas actividades correspondientes a su remediación. No hemos otorgado créditos retroactivos para las nuevas actividades en las mediciones pasadas. Esta es la razón por la que, por ejemplo, [CR3.4 Automatizar la detección de código malicioso] se informó como uno en BSIMM-V a pesar de que sabemos que un gran número de empresas participantes llevan a cabo esta actividad.

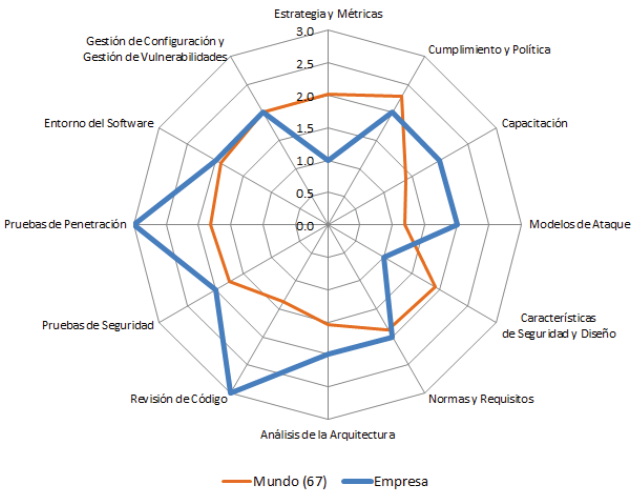
Nuestros criterios para añadir una actividad a BSIMM son los siguientes. Si observamos una actividad candidata que todavía no está en el modelo, determinamos cuántas empresas probablemente la llevan a cabo en base a los datos capturados previamente y a las consultas realizadas a través de la lista de correo BSIMM. Si la respuesta es "múltiples empresas", revisamos más detenidamente la actividad propuesta y nos hacemos una idea de cómo encaja en el modelo existente; si la respuesta es "una sola empresa", la actividad candidata se presenta como demasiado especializada. Además, si la actividad candidata está cubierta por actividades ya existentes, o simplemente refina o bifurca una actividad existente, se la descarta.

Utilizando los criterios anteriores, la actividad agregada al modelo BSIMM-V es: [CMVM3.4 Ejecutar un programa de recompensas para el descubrimiento de errores que impactan en la seguridad del software]. Incluimos una descripción completa de la nueva actividad en la descripción detallada BSIMM-V y en el Esqueleto BSIMM que le sigue.

Medición de su empresa con BSIMM-V

El uso más importante de BSIMM es como una vara de medición para determinar dónde se sitúa actualmente su enfoque en relación con otras empresas. Observe qué actividades ya tienen lugar, y utilice la "cobertura de actividad" para determinar sus niveles y construir una ficha de puntaje. En nuestro trabajo utilizando BSIMM para evaluar niveles, encontramos que el enfoque de representar la "marca de agua máxima" en el gráfico radial (basado en los tres niveles por práctica) es suficiente para obtener una sensación de baja resolución acerca de la madurez de la iniciativa, especialmente cuando se trabaja con datos de un segmento de industria vertical o localización geográfica particular.

Una comparación significativa es representar su propia marca de madurez usando la "marca de agua máxima" junto a los promedios que publicamos para ver cómo cuadra su iniciativa. A continuación, representamos los datos de una (falsa) empresa con relación al Mundo.



Ficha de Puntaje BSIMM-V para: EMPRESA **Puntaje Bruto: 37**

Gobernanza			Inteligencia			Puntos de Contacto con el SSDL			Despliegue		
Actividad	Empresas BSIMM-V	Empresa	Actividad	Empresas BSIMM.V	Empresa	Actividad	Empresas BSIMM-V	Empresa	Actividad	Empresas BSIMM-V	Empresa
[SM1.1]	44	1	[AM1.1]	21	1	[AA1.1]	56	1	[PT1.1]	62	1
[SM1.2]	34		[AM1.2]	43		[AA1.2]	35	1	[PT1.2]	51	1
[SM1.3]	34	1	[AM1.3]	30		[AA1.3]	24	1	[PT1.3]	43	
[SM1.4]	57	1	[AM1.4]	12	1	[AA1.4]	42		[PT2.2]	24	1
[SM1.6]	36		[AM1.5]	42	1	[AA2.1]	10		[PT2.3]	27	
[SM2.1]	26		[AM1.6]	16		[AA2.2]	8	1	[PT3.1]	13	1
[SM2.2]	31		[AM2.1]	7		[AA2.3]	20		[PT3.2]	8	
[SM2.3]	27		[AM2.2]	11	1	[AA3.1]	11				
[SM2.5]	20		[AM3.1]	4		[AA3.2]	4				
[SM3.1]	16		[AM3.2]	6							
[SM3.2]	6										
[CP1.1]	43	1	[SFD1.1]	54		[CR1.1]	24		[SE1.1]	34	
[CP1.2]	52		[SFD1.2]	53	1	[CR1.2]	34	1	[SE1.2]	61	1
[CP1.3]	45	1	[SFD2.1]	26		[CR1.4]	50	1	[SE2.2]	31	1
[CP2.1]	24		[SFD2.2]	29		[CR1.5]	23		[SE2.4]	25	
[CP2.2]	28		[SFD3.1]	9		[CR1.6]	25	1	[SE3.2]	10	
[CP2.3]	29		[SFD3.2]	13		[CR2.2]	10		[SE3.3]	9	
[CP2.4]	25		[SFD3.3]	9		[CR2.5]	15				
[CP2.5]	35	1				[CR2.6]	18				
[CP3.1]	14					[CR3.2]	4	1			
[CP3.2]	11					[CR3.3]	6				
[CP3.3]	8					[CR3.4]	1				
[T1.1]	50	1	[SR1.1]	48	1	[ST1.1]	51	1	[CMVM1.1]	59	1
[T1.5]	29		[SR1.2]	43		[ST1.3]	55	1	[CMVM1.2]	59	
[T1.6]	23	1	[SR1.3]	45	1	[ST2.1]	27	1	[CMVM2.1]	50	1
[T1.7]	33		[SR1.4]	27	1	[ST2.4]	13		[CMVM2.2]	44	
[T2.5]	9		[SR2.2]	23		[ST3.1]	11		[CMVM2.3]	30	
[T2.6]	13	1	[SR2.3]	19		[ST3.2]	8		[CMVM3.1]	6	
[T2.7]	9		[SR2.4]	19		[ST3.3]	8		[CMVM3.2]	6	
[T3.1]	4		[SR2.5]	22	1	[ST3.4]	6		[CMVM3.3]	2	
[T3.2]	4		[SR3.1]	8		[ST3.5]	7		*[CMVM3.4]*	*0*	
[T3.3]	8		[SR3.2]	12							
[T3.4]	9										
[T3.5]	5										

Leyenda	
Actividad	112 actividades BSIMM-V, que se organizan en 4 dominios y 12 prácticas
Empresas BSIMM	cantidad de empresas (hasta 67) observadas que ejecutan cada actividad
	la actividad más común dentro de una práctica
	una actividad común no observada en esta evaluación
	una actividad común observada en esta evaluación
	una práctica donde el puntaje de marca de agua superior está por debajo del promedio de BSIMM-V

Quizás el uso más obvio de BSIMM es realizar una comparación directa de las 112 actividades. Esto se puede lograr mediante la construcción de una ficha de puntajes con los datos que se muestran en la página 67.

La ficha de puntaje que usted ve aquí representa a una (falsa) empresa que realiza 37 actividades BSIMM (indicado con el valor 1 en las columnas "Empresa"), incluidas ocho actividades que son las más comunes en sus respectivas prácticas (celdas de color púrpura). Sin embargo, no realiza las actividades más comúnmente observadas en otras cuatro prácticas (celdas de color rojo), y la empresa deberá tomarse algún tiempo para determinar si estas actividades son necesarias o útiles para su iniciativa global de seguridad del software. La columna "Empresas BSIMM-V" muestra el número de observaciones (actualmente 67) para cada actividad, lo que permite que la empresa comprenda la popularidad general de una actividad entre los 67 participantes en BSIMM.

Una vez que usted haya determinado cuál es su opinión acerca de las actividades, puede elaborar un plan para mejorar sus prácticas con otras actividades sugeridas por BSIMM. Al proporcionar datos reales medidos en el terreno, BSIMM hace posible la creación de un plan a largo plazo para una iniciativa de seguridad del software así como el seguimiento de su progreso respecto a dicho plan. Que conste: no hay ninguna razón intrínseca para adoptar todas las actividades en todos los niveles para cada práctica. Adopte aquellas actividades que tengan sentido para su organización, e ignore aquellas que no lo tengan.

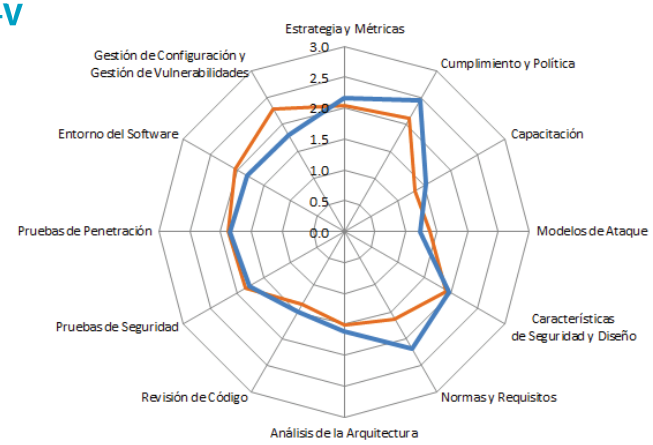
La mejor manera de utilizar BSIMM en la planificación es para identificar sus propias metas y objetivos, y recurrir a BSIMM para determinar qué actividades tienen sentido. No es recomendable optar por una u otra práctica. En particular, no creemos que una iniciativa exitosa pueda omitir todas las actividades de alguna de las 12 prácticas. Dicho de otra manera, los datos muestran que las iniciativas de alta madurez son bien equilibradas y llevan a cabo actividades de las 12 prácticas. En cualquier caso, explorar los objetivos y atender a lo que demanda su cultura es una manera mucho más limpia de proceder. Una vez que sepa cuáles son sus objetivos, puede determinar qué actividades adoptar. Tenga en cuenta que, en nuestra opinión, es mejor comenzar con algunas de las actividades del Nivel 1 de cada práctica en lugar de apresurarse a seleccionar actividades del Nivel 3, ignorando las demás. Este punto de vista está respaldado por los datos que hemos recogido.

El desglose de las actividades en niveles de madurez en cada práctica es sólo una guía. Los niveles proporcionan una progresión natural de las actividades asociadas con cada práctica. Sin embargo, no es en absoluto necesario llevar a cabo todas las actividades de un nivel determinado antes de pasar a las actividades de un nivel superior dentro de la misma práctica. Dicho lo anterior, identificamos los niveles por motivos de control estadístico. Las actividades del Nivel 1 (directas y sencillas) son las que se observan comúnmente; las del Nivel 2 (más difíciles y que requieren mayor coordinación), un poco menos; y las del Nivel 3 (muy complejas), raramente.

Al identificar los objetivos y las actividades de cada práctica que le serían adecuadas, y al garantizar un equilibrio apropiado respecto a los dominios, usted puede crear un plan estratégico para llevar adelante su iniciativa de seguridad del software. Tenga en cuenta que la mayoría de estas iniciativas son esfuerzos de varios años, respaldados por presupuestos, mandatos y dueños reales. Si bien todas las iniciativas se ven diferentes y se adaptan para ajustarse a una organización en particular, todas comparten, como describimos a continuación, actividades comunes.

Estudio de grupos de empresas en BSIMM-V

Los gráficos radiales que introdujimos anteriormente también son útiles para comparar grupos de empresas de industrias verticales o ubicaciones geográficas específicas. El siguiente gráfico muestra los datos de la industria vertical de servicios financieros -SFI- (26 empresas) y de proveedores independientes de software -PISW- (25 empresas), graficados juntos. En promedio, las empresas de servicios financieros tienen una madurez mayor a la de los proveedores independientes de software en 7 de las 12 prácticas. En la misma medida, los proveedores independientes de software tienen una madurez mayor a la de las



empresas de servicios financieros en 5 de las 12 prácticas. Si bien en este caso existe bastante solapamiento, las principales diferencias pueden explicarse con referencia al desplazamiento que se produce entre los proveedores independientes de software (ver CMVM y SE) hacia la computación en la nube, mientras que las empresas de servicios financieros tienden a enfatizar las prácticas de normas y cumplimiento (ver SR y CP). Una mirada más cercana a las 112 actividades revela diferencias más interesantes.

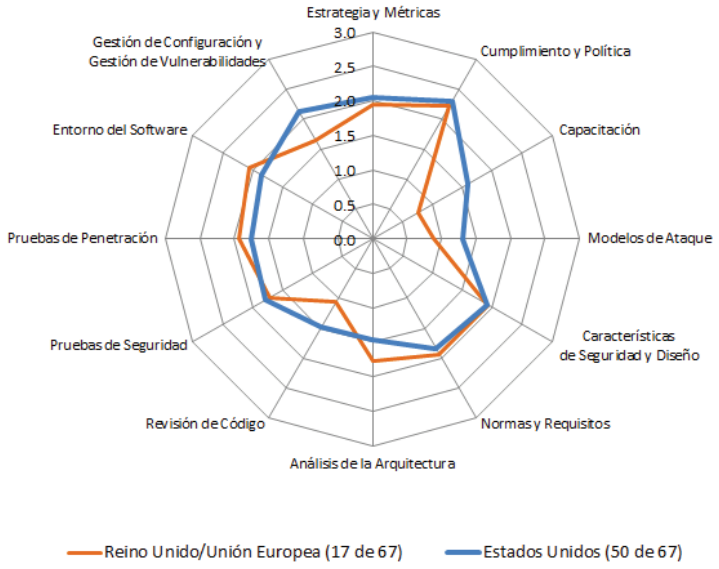
En la siguiente tabla puede ver la comparación de las Fichas de Puntaje BSIMM de las dos mayores industrias verticales. En las columnas "Actividad" destacamos en amarillo la actividad más común en cada práctica según lo observado en la totalidad de los datos BSIMM (67 empresas). La columna "Delta" muestra el valor absoluto de la diferencia entre las SFI y las PISW para cada actividad. Son de especial interés los valores delta grandes, que hemos destacado en azul oscuro cuando se observaron más PISW ejecutando la actividad, y en rojo cuando son más las SFI observadas ejecutándola.

Gobernanza				Inteligencia				Puntos de Contacto con el SSDL				Despliegue			
Actividad	SFI (de 26)	PISW (de 25)	Delta	Actividad	SFI (de 26)	PISW (de 25)	Delta	Actividad	SFI (de 26)	PISW (de 25)	Delta	Actividad	SFI (de 26)	PISW (de 25)	Delta
[SM1.1]	19	14	5	[AM1.1]	7	9	2	[AA1.1]	23	21	2	[PT1.1]	26	21	5
[SM1.2]	10	17	7	[AM1.2]	21	12	9	[AA1.2]	11	14	3	[PT1.2]	21	19	2
[SM1.3]	16	14	2	[AM1.3]	14	10	4	[AA1.3]	9	11	2	[PT1.3]	16	14	2
[SM1.4]	22	21	1	[AM1.4]	2	6	4	[AA1.4]	21	12	9	[PT2.2]	7	12	5
[SM1.6]	17	12	5	[AM1.5]	19	12	7	[AA2.1]	5	3	2	[PT2.3]	15	9	6
[SM2.1]	11	13	2	[AM1.6]	3	8	5	[AA2.2]	2	3	1	[PT3.1]	4	7	3
[SM2.2]	15	9	6	[AM2.1]	2	6	4	[AA2.3]	7	9	2	[PT3.2]	3	3	0
[SM2.3]	7	13	6	[AM2.2]	4	6	2	[AA3.1]	5	3	2				
[SM2.5]	10	7	3	[AM3.1]	1	3	2	[AA3.2]	0	3	3				
[SM3.1]	8	6	2	[AM3.2]	2	5	3								
[SM3.2]	0	3	3												
[CP1.1]	20	13	7	[SFD1.1]	21	21	0	[CR1.1]	9	9	0	[SE1.1]	15	12	3
[CP1.2]	20	19	1	[SFD1.2]	20	20	0	[CR1.2]	17	12	5	[SE1.2]	25	20	5
[CP1.3]	21	14	7	[SFD2.1]	10	9	1	[CR1.4]	19	19	0	[SE2.2]	12	13	1
[CP2.1]	12	8	4	[SFD2.2]	12	8	4	[CR1.5]	5	14	9	[SE2.4]	6	12	6
[CP2.2]	15	6	9	[SFD3.1]	4	3	1	[CR1.6]	10	9	1	[SE3.2]	2	5	3
[CP2.3]	8	11	3	[SFD3.2]	7	6	1	[CR2.2]	4	5	1	[SE3.3]	3	5	2
[CP2.4]	12	8	4	[SFD3.3]	3	6	3	[CR2.5]	7	4	3				
[CP2.5]	17	11	6					[CR2.6]	10	7	3				
[CP3.1]	12	0	12					[CR3.2]	3	1	2				
[CP3.2]	5	4	1					[CR3.3]	2	3	1				
[CP3.3]	2	5	3					[CR3.4]	1	0	1				
[T1.1]	21	17	4	[SR1.1]	21	17	4	[ST1.1]	22	18	4	[CMVM1.1]	24	22	2
[T1.5]	13	12	1	[SR1.2]	17	18	1	[ST1.3]	23	20	3	[CMVM1.2]	23	24	1
[T1.6]	6	11	5	[SR1.3]	16	16	0	[ST2.1]	11	12	1	[CMVM2.1]	20	20	0
[T1.7]	17	14	3	[SR1.4]	13	9	4	[ST2.4]	6	6	0	[CMVM2.2]	16	21	5
[T2.5]	3	3	0	[SR2.2]	12	7	5	[ST3.1]	3	6	3	[CMVM2.3]	9	14	5
[T2.6]	5	4	1	[SR2.3]	11	5	6	[ST3.2]	0	6	6	[CMVM3.1]	1	4	3
[T2.7]	1	5	4	[SR2.4]	4	9	5	[ST3.3]	1	4	3	[CMVM3.2]	1	4	3
[T3.1]	2	2	0	[SR2.5]	10	4	6	[ST3.4]	1	2	1	[CMVM3.3]	1	1	0
[T3.2]	2	1	1	[SR3.1]	2	5	3	[ST3.5]	2	4	2				
[T3.3]	1	4	3	[SR3.2]	9	2	7								
[T3.4]	5	4	1												
[T3.5]	2	2	0												

Existe un alto grado de superposición en las observaciones más comunes, con una mayoría de diferencias aproximándose a seis o menos. Sin embargo, en la práctica "Cumplimiento y política", hay cuatro actividades que se observan significativamente más a menudo en las SFI que en las PISW. Se trata de [CP1.1 Unificar las presiones regulatorias], [CP1.3 Crear una política], [CP2.2 Requerir la aprobación de seguridad para los riesgos relacionados con el cumplimiento] y [CP3.1 Crear documentación atractiva para reguladores]. El hecho de que las SFI pongan más énfasis en el cumplimiento y en las políticas que las PISW no resulta extraño teniendo en cuenta que las SFI están reguladas. Lo mismo ocurre con la clasificación de datos (las PISW producen cosas para almacenar datos y las SFI almacenan grandes cantidades de datos), lo que explica el delta considerable en [AM1.2 Crear un esquema de clasificación y un inventario de datos]. Por otro lado, las PISW tienden a apoyarse más en la persuasión que en la coerción al momento de influenciar a los equipos de desarrollo como queda reflejado por [SM1.2 Crear el rol de promotor y realizar el marketing interno], y también a aprovechar fuertemente la tecnología como se observa en [CR1.5 Hacer obligatoria la revisión de código para todos los proyectos].

Cuando comenzamos a trabajar en BSIMM, esperábamos datos que nos condujeran hacia explicaciones del comportamiento en industrias verticales o regiones geográficas específicas. No tuvimos tal suerte. Por ejemplo, cuando se trata de la seguridad del software, aunque existen pequeñas diferencias visibles entre el promedio de las SFI y el de las PISW, los puntos en común entre los participantes de alta madurez en cada industria vertical superan estas diferencias en un orden de magnitud.

Se puede decir lo mismo de las empresas de la Unión Europea + Reino Unido (17) y de los Estados Unidos (50). Las diferencias entre los grupos muestran un retraso en las iniciativas europeas respecto de las estadounidenses en doce a dieciocho meses (como se refleja en el gráfico radial que se muestra a continuación), y se corresponden con el desarrollo del mercado de la seguridad del software en estas regiones. Para un tratamiento en profundidad del mercado europeo, ver *BSIMM Europa: Measuring software security initiatives worldwide* <http://bit.ly/TTBfX>



BSIMM como un estudio longitudinal

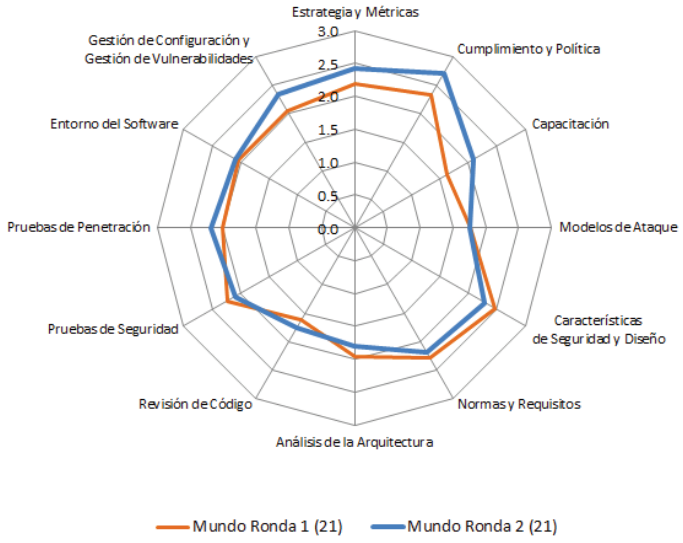
Veintiuna de las 67 empresas fueron medidas dos veces utilizando el sistema de medición BSIMM. En promedio, el intervalo de tiempo entre dos mediciones fue de 24 meses. Si bien cada una de las actividades de las doce prácticas va y viene, como se muestra en la Ficha de Puntaje Longitudinal que sigue, en general la remedición a lo largo del tiempo muestra una clara tendencia al aumento de la madurez en la población de las veintiuna empresas remedidas al momento.
El puntaje bruto en diecisiete de las veintiuna empresas aumentó un promedio de 14 (un aumento promedio del 16 %). Las iniciativas de seguridad del software maduran con el tiempo.

Gobernanza			Inteligencia			Puntos de Contacto con el SSDL			Despliegue		
Actividad	BSIMM Ronda 1 (de 21)	BSIMM Ronda 2 (de 21)	Actividad	BSIMM Ronda 1 (de 21)	BSIMM Ronda 2 (de 21)	Actividad	BSIMM Ronda 1 (de 21)	BSIMM Ronda 2 (de 21)	Actividad	BSIMM Ronda 1 (de 21)	BSIMM Ronda 2 (de 21)
[SM1.1]	13	20	[AM1.1]	8	8	[AA1.1]	17	20	[PT1.1]	20	20
[SM1.2]	15	13	[AM1.2]	14	15	[AA1.2]	15	13	[PT1.2]	14	19
[SM1.3]	14	17	[AM1.3]	11	13	[AA1.3]	15	13	[PT1.3]	13	15
[SM1.4]	18	21	[AM1.4]	10	7	[AA1.4]	12	15	[PT2.2]	8	9
[SM1.6]	15	15	[AM1.5]	13	17	[AA2.1]	8	7	[PT2.3]	8	9
[SM2.1]	11	13	[AM1.6]	7	12	[AA2.2]	3	3	[PT3.1]	6	8
[SM2.2]	10	13	[AM2.1]	9	6	[AA2.3]	9	9	[PT3.2]	5	5
[SM2.3]	13	10	[AM2.2]	7	10	[AA3.1]	5	6			
[SM2.5]	8	12	[AM3.1]	2	3	[AA3.2]	2	2			
[SM3.1]	5	9	[AM3.2]	2	5						
[SM3.2]	3	4									
[CP1.1]	15	15	[SFD1.1]	19	16	[CR1.1]	5	12	[SE1.1]	8	9
[CP1.2]	17	18	[SFD1.2]	12	18	[CR1.2]	12	16	[SE1.2]	19	18
[CP1.3]	19	19	[SFD2.1]	12	10	[CR1.4]	16	20	[SE2.2]	10	11
[CP2.1]	8	6	[SFD2.2]	10	13	[CR1.5]	9	10	[SE2.4]	8	12
[CP2.2]	13	11	[SFD3.1]	6	6	[CR1.6]	11	15	[SE3.2]	4	6
[CP2.3]	11	9	[SFD3.2]	6	5	[CR2.2]	9	7	[SE3.3]	5	2
[CP2.4]	7	13	[SFD3.3]	6	4	[CR2.5]	9	10			
[CP2.5]	13	16				[CR2.6]	5	12			
[CP3.1]	4	8				[CR3.2]	1	3			
[CP3.2]	4	6				[CR3.3]	2	3			
[CP3.3]	4	8				[CR3.4]	0	0			
[T1.1]	16	21	[SR1.1]	16	17	[ST1.1]	15	16	[CMVM1.1]	17	21
[T1.5]	10	16	[SR1.2]	12	19	[ST1.3]	11	15	[CMVM1.2]	17	20
[T1.6]	10	13	[SR1.3]	10	18	[ST2.1]	14	13	[CMVM2.1]	19	19
[T1.7]	10	18	[SR1.4]	9	11	[ST2.4]	6	7	[CMVM2.2]	13	19
[T2.5]	5	5	[SR2.2]	10	12	[ST3.1]	6	6	[CMVM2.3]	10	10
[T2.6]	4	7	[SR2.3]	7	8	[ST3.2]	9	7	[CMVM3.1]	3	3
[T2.7]	8	6	[SR2.4]	10	9	[ST3.3]	4	5	[CMVM3.2]	4	5
[T3.1]	3	4	[SR2.5]	7	12	[ST3.4]	3	3	[CMVM3.3]	0	0
[T3.2]	2	2	[SR3.1]	6	4	[ST3.5]	5	6			
[T3.3]	3	4	[SR3.2]	6	5						
[T3.4]	2	8									
[T3.5]	4	5									

Aquí tenemos dos maneras de pensar el cambio representado por la ficha de puntaje longitudinal. Vemos los cambios más grandes en actividades tales como [T1.7 Proporcionar capacitación individual a demanda], con 10 observaciones nuevas; [SM2.1 Publicar internamente datos sobre la seguridad del software] y [SR1.3 Traducir las restricciones de cumplimiento en requisitos], con 9 observaciones nuevas; y [SM2.5 Identificar métricas y utilizarlas para guiar los presupuestos], [CP2.4 Incluir en todos los contratos con proveedores acuerdos de nivel de servicio de seguridad del software] y [CR1.1 Crear una lista de los N principales errores de programación (preferir datos reales)], con 8 observaciones nuevas. Hay un adicional de cinco nuevas actividades observadas en siete empresas vueltas a medir.

Resulta menos obvio captar, a partir de la ficha de puntaje, la "variación" entre las actividades. Por ejemplo, mientras que el número de empresas se mantuvo constante para [CMVM2.3 Desarrollar un inventario operativo de las aplicaciones], cinco empresas tienen observaciones nuevas de esta actividad, pero ya no se la observa en otras cinco. Del mismo modo, hubo observaciones nuevas en cinco empresas para [T3.5 Establecer horarios de oficina del GSS] y [SR2.3 Crear normas para pilas de tecnología], mientras que se las dejó de observar en cuatro empresas. Además, hubo observaciones nuevas en cuatro empresas para [SFD3.2 Exigir la utilización de características de seguridad y marcos de referencia ya aprobados], mientras que se la dejó de observar en cinco empresas.

Utilizando el gráfico radial, podemos representar las marcas de agua máximas de la primera medición de las veintiuna empresas respecto de su segunda medición.



BSIMM a lo largo del tiempo

Esta es la quinta versión del proyecto BSIMM. El estudio original incluyó a nueve empresas y nueve mediciones distintas. BSIMM2 incluyó a 30 empresas y 42 mediciones diferentes (algunas empresas incluían filiales de gran tamaño, las que fueron medidas en forma independiente). BSIMM3 incluyó a 42 empresas, once de las cuales se volvieron a medir, para dar un total de 81 mediciones distintas. BSIMM4 incluyó 51 empresas, trece de las cuales se volvieron a medir (una de ellas, por tercera vez), lo que produjo un total de 95 mediciones distintas. BSIMM-V incluye 67 empresas, 21 de las cuales se han vuelto a medir (con cuatro empresas remedidas por tercera vez), lo que produjo un conjunto total de 161 mediciones diferentes. A partir de BSIMM-V, cinco empresas (que representaban 5 mediciones diferentes) han sido quitadas debido a que sus mediciones eran anteriores a 48 meses.

Comunidad BSIMM

Las 67 empresas participantes en el Proyecto BSIMM conforman la Comunidad BSIMM. Una lista de correo privada y moderada con más de 200 miembros le permite a los líderes de los GSS participar en BSIMM para discutir soluciones con otros que enfrentan los mismos problemas, discutir una estrategia con alguien que ya ha abordado un problema, buscar mentores entre aquellos que están más avanzados en su carrera profesional, y unirse entre sí para resolver problemas difíciles.

La Comunidad BSIMM también organiza conferencias anuales privadas, donde se reúnen hasta tres representantes de cada empresa en un foro extraoficial para discutir iniciativas de seguridad del software. A mediados del año 2012, 28 de 51 empresas participaron en la Tercera *BSIMM Community Conference* anual, llevada a cabo en Galloway, New Jersey. A principios del año 2013, 10 de las 15 empresas con presencia en Estados Unidos participaron de la Segunda *BSIMM Europe Community Conference* anual, celebrada en Londres, Inglaterra.

En su sitio web <http://bsimm.com> BSIMM incluye una sección para usuarios registrados integrantes de la comunidad BSIMM, donde se publica información acerca de conferencias, grupos de trabajo y estudios iniciados a través de la lista de correo.

Modelo de Referencia para la Seguridad del Software

La tabla que sigue muestra el MRSS. Hay 12 prácticas organizadas dentro de cuatro *dominios*. Los dominios son:

1. **Gobernanza:** Prácticas que ayudan a organizar, administrar y medir una iniciativa de seguridad del software. El desarrollo del equipo también es una práctica fundamental de gobernanza.
2. **Inteligencia:** Prácticas que dan lugar a una recopilación de conocimiento corporativo que se utiliza para llevar a cabo las actividades de seguridad del software en toda la organización. Esta recopilación incluye tanto la orientación para una seguridad proactiva como el modelado de amenazas desde la propia organización.
3. **Puntos de Contacto con el SSDL (*SSDL Touchpoints*):** Prácticas asociadas con el análisis y la garantía de los artefactos y procesos de desarrollo de software. Todas las metodologías de seguridad del software incluyen estas prácticas.
4. **Despliegue:** Prácticas en las que se interactúa con los grupos tradicionales de seguridad de redes y de mantenimiento del software. La configuración, el mantenimiento y otros elementos del entorno del software tienen impacto directo sobre la seguridad de éste.

Modelo de Referencia para la Seguridad del Software (MRSS)

Gobernanza	Inteligencia	Puntos de Contacto con el SSDL	Despliegue
Estrategia y Métricas	Modelos de Ataque	Análisis de la Arquitectura	Pruebas de Penetración
Cumplimiento y Política	Características de Seguridad y Diseño	Revisión de Código	Entorno del Software
Capacitación	Normas y Requisitos	Pruebas de Seguridad	Gestión de Configuración y Gestión de Vulnerabilidades

Existen tres prácticas dentro de cada dominio. Para dar una idea de qué implica una práctica, incluimos una breve explicación de cada una. Cabe destacar que BSIMM describe los objetivos y actividades para cada práctica.

En el dominio **Gobernanza**, la práctica *Estrategia y Métricas* abarca la planificación, la asignación de roles y responsabilidades, la identificación de las metas de seguridad del software, la determinación de los presupuestos, y la identificación de métricas y puertas de control (gates). La práctica *Cumplimiento y Política* se centra en la identificación de los controles para el cumplimiento de marcos regulatorios, tales como PCI DSS y HIPAA;

el desarrollo de controles contractuales, tales como los acuerdos de nivel de servicio para ayudar a controlar el riesgo del software comercial enlatado; la definición de la política de seguridad del software de la organización; y la auditoría con respecto a dicha política. La práctica *Capacitación* siempre desempeñó un papel fundamental en la seguridad del software, ya que los desarrolladores y los arquitectos de software a menudo comienzan con muy pocos conocimientos acerca de seguridad.

Las prácticas en el dominio **Inteligencia** tienen la finalidad de crear recursos propios dentro de la organización. Estos recursos están divididos en tres prácticas. La práctica *Modelos de Ataque* captura información que es utilizada para pensar como un atacante: modelado de amenazas, desarrollo y refinamiento de casos de abuso, clasificación de datos y patrones de ataque específicos de la tecnología. La práctica *Características de Seguridad y Diseño* se encarga de la creación de patrones de seguridad que se puedan utilizar para los controles de seguridad más importantes (que cumplen con las normas definidas en la siguiente práctica), la construcción de marcos de referencia de middleware para dichos controles, y la creación y publicación de otras directrices de seguridad proactiva. La práctica *Normas y Requisitos* consiste en la obtención de los requisitos explícitos de seguridad de la organización, la determinación de qué paquetes de software comercial enlatado recomendar, la construcción de normas para los controles de seguridad más importantes (tales como autenticación, validación de entrada, y así sucesivamente), la creación de normas de seguridad para las tecnologías en uso, y la creación de un comité de revisión de normas.

El dominio **Puntos de Contacto con el SSDL** probablemente es el más conocido de los cuatro. Este dominio incluye las mejores prácticas esenciales de seguridad del software que se integran en el SDLC. Las dos prácticas de seguridad del software más importantes son el Análisis de la Arquitectura y la Revisión de Código. La práctica *Análisis de la Arquitectura* abarca la captura de la arquitectura del software en diagramas concisos, la aplicación de listas de riesgos y amenazas, la adopción de un proceso de revisión (como STRIDE o *Architectural Risk Analysis*) y la construcción de un plan de evaluación y correcciones para la organización. La práctica *Revisión de Código* incluye el uso de herramientas para comprobar el código, el desarrollo de reglas a medida, los perfiles personalizados para el uso de aplicaciones por parte de diferentes roles (por ejemplo, desarrolladores versus auditores), el análisis manual, y los resultados de seguimiento/medición. La práctica *Pruebas de Seguridad* se refiere a las verificaciones previas al lanzamiento, incluida la integración de la seguridad dentro de los procesos estándares de aseguramiento de calidad. La práctica incluye el uso de herramientas de seguridad de caja negra (incluidas las pruebas *fuzzing*) tales como una prueba de humo en QA, las pruebas de caja blanca guiadas por riesgos, la aplicación del modelo de ataque y el análisis de cobertura de código. Las pruebas de seguridad se centran en vulnerabilidades en el desarrollo.

Por el contrario, en el dominio **Despliegue** la práctica *Pruebas de Penetración* implica pruebas realizadas desde un punto de vista externo, como las que típicamente llevan a cabo especialistas de seguridad informática. Las pruebas de penetración se centran en las vulnerabilidades de la configuración final, y retroalimentan directamente a la gestión con información sobre defectos y mitigación. La práctica *Entorno del Software* se ocupa de la aplicación de parches del sistema operativo y de la plataforma, de los firewalls de aplicaciones web, de la documentación de instalación y configuración, de la monitorización de aplicaciones, de la gestión de cambios, y, en última instancia, de la firma del código. Por último, la práctica *Gestión de Configuración y Gestión de Vulnerabilidades* se ocupa de la aplicación de parches y actualizaciones, del control de versiones, del seguimiento de defectos y las respectivas correcciones, y del manejo de incidentes.

BSIMM-V

Presentamos el modelo de madurez como una serie de actividades asociadas con cada una de las doce prácticas. Nuestro enfoque consiste en identificar las metas para cada nivel de una práctica, metas que se pueden subdividir en objetivos de la práctica/nivel y que están, de esta manera, asociados con las actividades. Si necesita ayuda para entender cómo están organizadas las actividades básicas, consulte el esquema BSIMM.

Caracterizar y "vender" las metas de negocio del proyecto es parte del éxito de una iniciativa de seguridad del software. Desde una perspectiva descendente, un enfoque basado en metas incluye la identificación de las mismas a nivel de la iniciativa y a nivel del dominio, así como las metas y objetivos de las prácticas, niveles y actividades.

A riesgo de repetirnos, las metas de alto nivel más notables e importantes para BSIMM son las siguientes:

- Decisiones fundamentadas para la gestión de riesgo
- Claridad acerca de qué es "hacer lo correcto" para cada uno de los involucrados en la seguridad del software
- Reducción de costos a través de procesos estándares y repetibles
- Mejora de la calidad del código

Cada uno de los cuatro dominios del MRSS tiene asociadas metas particulares. Al comprender estas metas, usted podrá ver rápidamente por qué tiene sentido adoptar algunos aspectos de todos los dominios. Ciertamente, ignorar todo un dominio sería insensato. Estas son las metas asociados a cada dominio de BSIMM:

Dominio	Metas de Negocio
Gobernanza	Transparencia. Responsabilidad. Verificaciones y balances
Inteligencia	Auditoría. Diligencia. Estandarización
Puntos de Contacto con el SSDL	Control de calidad
Despliegue	Control de calidad. Gestión de cambios

En el MRSS hay tres prácticas en cada dominio. Al igual que los dominios, cada práctica tiene una meta de alto nivel. Al comprender estas metas, usted puede entender fácilmente por qué tiene sentido adoptar algunos aspectos de todas las prácticas. En ese momento, puede comenzar a pensar en cuáles son las metas más importantes para su organización y su cultura. Estas son las metas asociadas a cada práctica en BSIMM:

Dominio	Práctica	Metas de Negocio
Gobernanza	Estrategia y Métricas	Transparencia en las expectativas. Responsabilidad por los resultados.
	Cumplimiento y Política	Guía prescriptiva para todos los interesados. Auditoría.
	Capacitación	Personal capacitado. Corrección de errores.
Inteligencia	Modelos de Ataque	Conocimiento personalizado.
	Características de Seguridad y Diseño	Diseños reutilizables. Guía prescriptiva para todos los interesados.
	Normas y Requisitos	Guía prescriptiva para todas las partes interesadas
Puntos de Contacto con el SSDL	Análisis de la Arquitectura	Control de calidad
	Revisión de Código	Control de calidad
	Pruebas de Seguridad	Control de calidad
Despliegue	Pruebas de Penetración	Control de calidad
	Entorno del Software	Gestión de cambios
	Gestión de Configuración y Gestión de Vulnerabilidades	Gestión de cambios

GOBERNANZA: Estrategia y Métricas (SM)

Las metas generales de la práctica Estrategia y Métricas son la transparencia en las expectativas y la responsabilidad por los resultados. La dirección ejecutiva debe clarificar las expectativas de la organización para el SSDL de manera que todos comprendan la importancia de la iniciativa. Además, debe establecer objetivos específicos para todas las partes interesadas en el SSDL y establecer que determinadas personas se hagan responsables de cumplir con esos objetivos.

uno

SM Nivel 1: Lograr una comprensión común de la dirección y la estrategia. Los administradores deben asegurarse de que todas las personas relacionadas con la creación, el despliegue, la operación y el mantenimiento del software comprendan los objetivos de seguridad determinados por la organización. Los líderes también deben asegurarse de que la organización en su conjunto comprenda la estrategia para alcanzar estos objetivos. Una comprensión estratégica común es esencial para la ejecución efectiva y eficiente del programa.

[SM1.1] Publicar el proceso (roles, responsabilidades, plan), evolucionar a medida que sea necesario. El proceso para abordar la seguridad del software se difunde entre todos los participantes para que conozcan el plan. Se definen explícitamente las metas, los roles, las responsabilidades y las actividades. La mayoría de las organizaciones seleccionan alguna metodología publicada, como Microsoft SDL o Touchpoints de Cigital, y luego la adaptan a sus necesidades. Un proceso SSDL evoluciona a medida que la organización madura y cambia el panorama de la seguridad. En muchos casos, la metodología sólo se publica internamente y el GSS la controla. No es necesario que el SSDL sea promocionado fuera de la empresa para que tenga valor.

[SM1.2] Crear el rol de promotor y realizar el marketing interno. Las conversaciones ad hoc entre el GSS y los ejecutivos o un GSS donde "todo el mundo es un promotor" no obtienen los resultados deseados. Por eso, con el fin de lograr que toda la organización apoye la iniciativa de seguridad del software, algún integrante del GSS desempeña el rol de promotor. Dicha función ayuda a mantener a la organización al tanto de la magnitud del problema de la seguridad del software y de las acciones necesarias para su solución. Los promotores pueden dar charlas a grupos internos, invitar a oradores externos, redactar guías ejecutivas para consumo interno, o crear una colección de documentos, libros y otros recursos en un sitio web interno y promover su uso. El rol de Michael Howard en Microsoft es un ejemplo clásico del tipo de promotor sugerido.

[SM1.3] Educar a los ejecutivos. Los ejecutivos aprenden de las consecuencias de una inadecuada seguridad del software y del impacto comercial negativo que puede tener una seguridad pobre. También se informan sobre qué están haciendo otras organizaciones para lograr la seguridad de su software. Al entender cuál es el problema y su solución adecuada, los ejecutivos apoyan la iniciativa de seguridad del software como una necesidad de gestión de riesgo. En su forma más peligrosa, tal educación llega gracias a la acción de hackers maliciosos o a incidentes de exposición pública de datos. Preferentemente, el GSS monta un escenario para representar el peor de los casos posibles, en un ambiente controlado y con el permiso de todos los involucrados (por ejemplo, muestra cómo funcionan realmente los ataques a una vulnerabilidad y su impacto en el negocio). En algunos casos, esta presentación a los ejecutivos de más alto nivel puede ayudar a obtener recursos para una iniciativa de seguridad del software en curso. A menudo, cuando se trata de reforzar la atención ejecutiva, resulta de utilidad traer a un gurú externo.

[SM1.4] Identificar la ubicación para puertas de control (*gates*), reunir los artefactos necesarios. El proceso de seguridad del software incluirá puertas de control de lanzamiento/puntos de verificación/hitos en uno o más puntos dentro del SDLC o, más probablemente, de varios SDLC. Los dos primeros pasos hacia el establecimiento de las puertas de control de lanzamiento son: 1) identificar ubicaciones para puertas de control compatibles con las prácticas de desarrollo existentes y 2) comenzar a reunir la información necesaria para tomar una decisión de tipo "pasa o no pasa". Es importante destacar que, en este nivel, las puertas de control no son de cumplimiento obligatorio. Por ejemplo, el GSS puede recolectar los resultados de las pruebas de seguridad de cada proyecto antes de su lanzamiento, sin llegar a emitir un juicio sobre qué constituye una prueba suficiente o resultados aceptables de las pruebas. La idea de primero identificar las puertas de control y más tarde hacerlas obligatorias es de gran ayuda para avanzar en el desarrollo de la seguridad del software sin grandes sufrimientos.

Difundir cuáles son las puertas de control y activarlas solo una vez que la mayoría de los proyectos ya saben cómo tener éxito. Este enfoque gradual sirve para motivar el buen comportamiento sin necesidad de requerirlo.

[SM1.6] Requerir la aprobación de la seguridad. La organización cuenta con un proceso para la aceptación de los riesgos de seguridad y documentación de responsabilidades, que abarca a toda la iniciativa. Un responsable de aceptar el riesgo aprueba el estado de todo el software previo a su lanzamiento. Por ejemplo, la política de aprobación podría requerir que el jefe de la unidad de negocio apruebe la existencia de vulnerabilidades críticas que no se mitigaron o que se hayan omitido pasos del SSDL. Una aceptación del riesgo que se hace de manera informal no cuenta como una aprobación de la seguridad; el acto de aceptación es más efectivo cuando está formalizado (por ejemplo, con una firma, la presentación de un formulario o similar) y se lo captura como referencia futura.

dos

SM Nivel 2: Alinear el comportamiento con la estrategia y verificar la adhesión. Los gerentes deben identificar explícitamente a aquellos que tienen a su cargo la responsabilidad de la gestión de riesgos de la seguridad del software. Estas personas, a su vez, son responsables de garantizar el desempeño exitoso de las actividades del SSDL. Los administradores del SSDL deben garantizar una rápida identificación y modificación de cualquier comportamiento del SSDL que resulte en un riesgo inaceptable. Para reducir el riesgo inaceptable, los gerentes deben identificar y estimular el crecimiento de un satélite de seguridad del software (ver la sección previa "Roles").

[SM2.1] Publicar internamente datos sobre la seguridad del software. El GSS publica internamente datos sobre el estado de la seguridad del software en la organización. La información se puede brindar como un tablero de mando con métricas para ejecutivos y gerentes de desarrollo de software. A veces, la publicación no se comparte con todos los miembros de una empresa, sino solo con los ejecutivos relevantes: alcanza con transmitir la información a aquellos ejecutivos que luego harán algo al respecto y conducirán los cambios en la organización. En otros casos, la gestión "a libro abierto" y la transmisión de los datos a quienes estén interesados ayudan a que todos sepan lo que está sucediendo, con la filosofía de que "la luz del sol es el mejor desinfectante". Si la cultura de la organización promueve la competencia interna entre los grupos, esta información añade una dimensión al juego, la de la seguridad.

[SM2.2] Cumplir obligatoriamente con las puertas de control satisfaciendo mediciones y realizar el seguimiento de las excepciones. Ahora las puertas de control de seguridad del SDLC son de cumplimiento obligatorio: para atravesar una puerta de control, un proyecto debe cumplir con una medición establecida u obtener una exención. Incluso los equipos de proyecto más reacios deben ahora cumplir con estos requisitos. El GSS realiza el seguimiento de las excepciones. Una puerta de control podría requerir que un proyecto se someta a una revisión de código y remedie cualquier hallazgo crítico antes de su lanzamiento. En algunos casos, las puertas de control están directamente asociadas con los controles requeridos por regulaciones, acuerdos contractuales y otras obligaciones del negocio, y entonces el seguimiento de las excepciones se realiza conforme a los requerimientos legales o regulatorios. En otros casos, las mediciones de las puertas de control producen indicadores clave de rendimiento que se utilizan para gobernar el proceso.

[SM2.3] Crear o hacer crecer un satélite. El satélite comienza como un conjunto de personas diseminadas a lo largo de la organización que muestran un nivel de interés o habilidad respecto de la seguridad por encima de la media. La identificación de este grupo es un paso hacia la creación de una red social que acelere la adopción de la seguridad en el desarrollo de software. Una forma de empezar es hacer un seguimiento de las personas que se destacaron durante los cursos de capacitación introductoria. (Véase [T2.7 Identificar el satélite a través de la capacitación]). Otra manera es solicitar voluntarios. En un enfoque más verticalista, la membresía del satélite inicial se asigna para garantizar una cobertura completa de todos los grupos de desarrollo/producto. La membresía definitiva se debe basar en el desempeño real. Un satélite fuerte es una buena señal de una iniciativa madura de seguridad del software.

[SM2.5] Identificar métricas y utilizarlas para guiar los presupuestos. El GSS y su equipo de gestión eligen las métricas que definen y miden el progreso de la iniciativa de seguridad del software. Estas métricas

guiarán el presupuesto de la iniciativa y la asignación de los recursos, por lo que no son suficientes cuentas y estadísticas simples. Las métricas también permitirán que el GSS explique sus metas y su progreso en términos cuantitativos. Una métrica de este tipo podría ser la densidad de defectos de seguridad. Una reducción en dicha densidad podría utilizarse para mostrar un costo de corrección decreciente a través del tiempo. La clave aquí es vincular los resultados técnicos con los objetivos del negocio de una manera clara y evidente para justificar el financiamiento. Dado que el concepto de seguridad resulta de por sí poco convincente para las personas vinculadas al negocio, puede resultar muy útil hacer explícita esta vinculación.

tres

SM Nivel 3: Poner en práctica la gestión de la cartera basada en riesgos. Los dueños de aplicaciones y el GSS deben informar a la gerencia del riesgo asociado a cada aplicación de la cartera. El GSS debe anunciar sus actividades externamente para generar respaldo a su enfoque y posibilitar la seguridad del ecosistema.

[SM3.1] Utilizar una aplicación de seguimiento interno con una visión de cartera. El GSS utiliza una aplicación de seguimiento centralizado para medir el progreso de cada porción de software en su ámbito. La aplicación registra las actividades de seguridad programadas, en progreso y completadas. Incorpora los resultados de actividades tales como el análisis de la arquitectura, la revisión de código y las pruebas de seguridad. El GSS utiliza la aplicación de seguimiento para generar informes de cartera para muchas de las métricas que utiliza. Es fundamental lograr una vista combinada, de inventario y de postura frente al riesgo. En muchos casos, estos datos se difunden al menos entre los ejecutivos. Dependiendo de la cultura empresarial, esto puede causar efectos interesantes gracias a la competencia interna. A medida que una iniciativa madura y las actividades se vuelven más distribuidas, el GSS irá utilizando el sistema de informes centralizado para realizar un seguimiento de todas las partes en movimiento.

[SM3.2] Ejecutar un programa de marketing externo. El GSS promociona la iniciativa de seguridad del software fuera de la empresa para construir un respaldo externo. La seguridad del software va más allá de ser un ejercicio de reducción de riesgos y se convierte en una ventaja competitiva o un diferenciador en el mercado. El GSS puede escribir artículos o libros, tener un blog público o participar en conferencias externas o exposiciones comerciales. En algunos casos, se puede publicar una metodología SSDL completa y para luego promoverla externamente. Compartir detalles con el exterior y promover su crítica puede aportar nuevas perspectivas dentro de la empresa.

GOBERNANZA: Cumplimiento y Política (CP)

Las metas generales de la práctica Cumplimiento y Política son la orientación prescriptiva para todos los interesados y la auditoría de las actividades del SSDL. La orientación prescriptiva aprobada por la gerencia debe estar disponible para todos los interesados en el SSDL, incluidos los proveedores, para que pueda así ser utilizada en la satisfacción de los objetivos de seguridad y cumplimiento. Todas las actividades del SSDL deben producir artefactos suficientes para verificar la adhesión a la orientación prescriptiva.

uno

CP Nivel 1: Documentar y unificar el cumplimiento estatutario, regulatorio y contractual. El GSS debe trabajar con grupos adecuados para capturar, de forma unificada, los requisitos de cumplimiento en la orientación prescriptiva, y luego debe poner ese conocimiento a disposición de las partes interesadas en el SSDL.

[CP1.1] Unificar las presiones regulatorias. Si el negocio o sus clientes están sujetos a requisitos regulatorios o de cumplimiento, tales como FFIEC, GLBA, OCC, PCI DSS, SOX, HIPAA u otros, el GSS actúa como centro focal encargado de comprender las restricciones que dichos requisitos imponen al software. En algunos casos, el GSS crea un enfoque unificado que elimina redundancias debidas a la superposición de estas exigencias. Un enfoque formal asignará porciones de las regulaciones aplicables a declaraciones de controles que explican cómo la organización cumple con ellas. Como una alternativa, los procesos de negocios existentes que se ejecutan en el área de Asuntos Legales o en la de Gestión de Riesgo y Cumplimiento externas al GSS también pueden servir como puntos focales para los requisitos regulatorios. La meta de esta actividad es crear un conjunto de orientaciones de seguridad del software de tal manera que el cumplimiento se complete tan eficientemente como sea posible (en su mayoría, gracias a la eliminación de duplicados). Algunas empresas progresan en su exposición pública involucrándose directamente en grupos de normalización con el fin de influir en el ambiente regulatorio.

[CP1.2] Identificar obligaciones relativas a la información de identificación personal (IIP). La forma en que el software maneja la IIP puede estar explícitamente regulada, pero incluso si no lo está, la privacidad es un tema candente. El GSS asume un rol de liderazgo en la identificación de las obligaciones relativas a la IIP derivadas del marco regulatorio y de las expectativas del cliente. Utiliza esta información para promover mejores prácticas relacionadas con la privacidad. Por ejemplo, si la organización procesa transacciones de tarjetas de crédito, el GSS identificará las restricciones que PCI DSS impone en el tratamiento de los datos del titular de tarjeta. Hay que tener en cuenta que la tercerización hacia ambientes de hospedaje (por ejemplo, "la nube") no relaja la mayoría de las obligaciones relacionadas con la IIP. También se debe reparar en que las empresas que crean productos de software para procesar IIP (pero que no la manejan necesariamente en forma directa) pueden proporcionar controles de privacidad y orientación a sus clientes.

[CP1.3] Crear una política. El GSS guía al resto de la organización creando la política de seguridad del software que cumpla con los requisitos de seguridad regulatorios y del cliente, o bien contribuyendo a su creación. La política ofrece un enfoque unificado cuya observancia permite satisfacer la lista (potencialmente larga) de controles de seguridad a nivel de Gobernanza. Como resultado, los equipos de proyecto pueden evitar tener que aprender los detalles relacionados con el cumplimiento de todas las regulaciones aplicables. Del mismo modo, no tienen necesidad de volver a aprender por sí mismos los requisitos de seguridad del cliente. Los documentos de políticas del GSS a veces se focalizan en temas cuyo cumplimiento resulta muy importante, tales como el manejo de la IIP o el uso de criptografía; en algunos otros casos, los documentos de políticas se relacionan directamente con el SSDL y su uso dentro de la empresa. Las normas de arquitectura y las directrices de codificación no son ejemplos de una política de seguridad del software. Por otro lado, una política que prescribe y exige el uso de las directrices de codificación y las normas de arquitectura para ciertas categorías de aplicaciones sí lo es. La política es lo que está permitido y lo que está denegado a nivel de la iniciativa.

dos

CP Nivel 2: Alinear las prácticas internas con los controles de cumplimiento y la política, avalados por los ejecutivos. Los ejecutivos deben promover abiertamente al GSS y la iniciativa de seguridad del software asociada, incluida la necesidad de su cumplimiento. Los administradores del riesgo deben asumir explícitamente la responsabilidad por el riesgo del software. El GSS y los dueños de las aplicaciones deben asegurarse de que los acuerdos de nivel de servicio de los proveedores de software contemplen propiedades de seguridad de sus entregables.

[CP2.1] Identificar el inventario de datos de IIP. La organización identifica los tipos de IIP almacenados por cada uno de sus sistemas y sus repositorios de datos. Un inventario de IIP se puede abordar de dos maneras: relevando cada aplicación individual e indicando qué IIP usa, o clasificando tipos particulares de IIP y luego las aplicaciones que los manipulan. Cuando se lo combina con las obligaciones de la organización relativas a la IIP, este inventario guía la planificación de privacidad. Por ejemplo, el GSS ahora puede crear una lista de las bases de datos que requieren que se notifique al cliente si sufren una brecha de seguridad.

[CP2.2] Requerir la aprobación de seguridad para los riesgos relacionados con el cumplimiento. La organización cuenta con un proceso formal para la aceptación y responsabilidad por los resultados de los riesgos de cumplimiento aplicable a todos los proyectos de desarrollo de software. El responsable de aceptar el riesgo aprueba el estado del software previo a su lanzamiento. Por ejemplo, la política de aprobación puede requerir que el jefe de la unidad de negocio apruebe que no se hayan mitigado cuestiones de cumplimiento o que se hayan omitido pasos del SSDL relacionados con el cumplimiento. La aprobación debería ser explícita y capturada como referencia futura, y se debería realizar un seguimiento de las excepciones.

[CP2.3] Implementar y realizar el seguimiento de los controles para el cumplimiento. La organización puede demostrar el cumplimiento de las regulaciones aplicables ya que sus prácticas están alineadas con las declaraciones de control desarrolladas por el GSS. (Ver [CP1.1 Unificar las presiones regulatorias]). El GSS realiza el seguimiento de los controles, presta asistencia a las áreas problemáticas y garantiza la satisfacción de los auditores y los organismos reguladores. Si el SDLC de la organización es predecible y fiable, el GSS puede cruzarse de brazos y observar. Si el SDLC es variable o menos fiable, el GSS se puede ver obligado a tomar un rol más activo como árbitro. Una empresa que realiza esta actividad correctamente puede vincular de manera explícita su SSDL con la satisfacción de sus asuntos de interés relativos al cumplimiento.

[CP2.4] Incluir en todos los contratos con proveedores acuerdos de nivel de servicio de seguridad del software (SLA). Los contratos de proveedores incluyen un SLA para garantizar que estos no pondrán en peligro el historial de cumplimiento y la iniciativa de seguridad del software de la organización. Cada contrato nuevo o renovado contiene un conjunto de disposiciones que obligan al proveedor a entregar un producto o servicio compatible con la política de seguridad de la organización (véase [SR2.5 Crear un modelo de SLA]). En algunos casos, las preocupaciones sobre el uso de software con licencia de código abierto dan inicio al proceso de control de los proveedores. Esto puede abrir la puerta a un lenguaje ampliado de la seguridad del software dentro del SLA.

[CP2.5] Promover la conciencia ejecutiva acerca de las obligaciones de cumplimiento y privacidad. El GSS logra la aceptación ejecutiva en cuanto a las actividades de cumplimiento y privacidad. Se les proveen a los ejecutivos explicaciones en un lenguaje sencillo acerca de las obligaciones de cumplimiento y privacidad de la organización y de las potenciales consecuencias de no cumplir con dichas obligaciones. Para algunas organizaciones, una manera eficaz de abordar el tema puede ser explicar el costo directo y las consecuencias probables de una pérdida de datos. Para otras organizaciones, da resultado una presentación de expertos externos al directorio, dado que algunos ejecutivos valoran la perspectiva externa por sobre la interna. Una señal segura de que se realizó una concientización apropiada de los ejecutivos es la asignación adecuada de recursos para hacer el trabajo.

tres

CP Nivel 3: Conducir la evolución de las políticas y las demandas a los proveedores en base a datos acerca de amenazas, ataques, defectos y problemas operativos de la organización. Los ejecutivos deben garantizar que la política de seguridad del software se actualiza periódicamente utilizando datos reales, y deben demostrar el cumplimiento continuo por parte de la organización. El GSS, los dueños de aplicaciones y los grupos de las áreas legales deben garantizar que los proveedores entregan software que cumple con la política pertinente de la organización.

[CP3.1] Crear documentación atractiva para reguladores. El GSS posee la información que necesitan los reguladores. La combinación de una política escrita, una documentación de controles y unos artefactos recogidos a través del SSDL le brinda al GSS la capacidad de demostrar el historial de cumplimiento de la organización sin tener que pasar por un "simulacro de incendio" en cada auditoría. En algunos casos, los reguladores, auditores y la alta dirección quedan satisfechos con los mismos tipos de informes, que se pueden generar directamente empleando diferentes herramientas.

[CP3.2] Imponer políticas a proveedores. Los proveedores están obligados a adherir a las mismas políticas de uso interno. Deben presentar evidencias de que sus prácticas de seguridad del software son aceptables. Las evidencias pueden incluir resultados de revisión de código o pruebas de penetración. Los proveedores también pueden dar testimonio del hecho de que están llevando a cabo ciertos procesos SSDL. En algunos casos, se puede utilizar un puntaje BSIMM o vBSIMM para ayudar a garantizar que los proveedores cumplen con las políticas de la empresa.

[CP3.3] Conducir la retroalimentación de datos desde el SSDL hacia las políticas. La información del SSDL habitualmente realimenta el proceso de creación de políticas. Las políticas se mejoran para detectar en forma temprana defectos o evitar que ocurran en primer lugar. Se eliminan los puntos ciegos en base a las tendencias que revelan las fallas del SSDL; por ejemplo, el análisis inadecuado de la arquitectura, vulnerabilidades recurrentes, puertas de control de seguridad ignoradas o la elección de la empresa equivocada para llevar a cabo una prueba de penetración pueden poner de manifiesto la debilidad de las políticas. Con el tiempo, éstas se deberían volver más prácticas y fáciles de llevar a cabo (véase [SM1.1 “Publicar el proceso (roles, responsabilidades, plan), evolucionar a medida que sea necesario”]). Al final, las políticas se alinean a sí mismas con los datos del SSDL y así aumentan y mejoran la eficacia de la empresa.

GOBERNANZA: Capacitación (T)

Las metas generales de la práctica Capacitación son la creación de una fuerza de trabajo avezada y la corrección de errores en los procesos. La fuerza de trabajo debe contar con conocimientos basados en roles que incluyen, específicamente, las competencias necesarias para realizar de manera adecuada sus actividades a lo largo del SSDL. La capacitación debe incluir información específica sobre las causas de fondo de los errores descubiertos en las actividades y los resultados del proceso.

uno

T Nivel 1: Crear una capacitación personalizada, basada en roles y disponible bajo demanda. El GSS debe despertar el interés en la seguridad del software en toda la organización y proporcionar material de capacitación específico basado en roles, incluida la capacitación asistida por computadora, que incorpora lecciones tomadas de eventos internos reales.

[T1.1] Proveer capacitación orientada a la concientización. El GSS provee capacitación orientada a la concientización con el fin específico de promover una cultura de la seguridad del software en toda la organización. La capacitación puede ser provista por los miembros del GSS, una empresa externa o el área de capacitación interna de la organización, o puede realizarse mediante un sistema asistido por computadora. El contenido de la capacitación no está necesariamente adaptado a una audiencia específica; por ejemplo, todos los programadores, ingenieros de aseguramiento de la calidad y administradores de proyecto pueden asistir al mismo curso introductorio sobre seguridad del software. Esta actividad común se puede mejorar con un curso introductorio hecho a medida, cuyo enfoque responda explícitamente a la cultura de la empresa. Los cursos introductorios genéricos que tratan aspectos básicos de seguridad en TI y conceptos de alto nivel sobre seguridad del software no generan resultados satisfactorios. Del mismo modo, concientizar solo a desarrolladores y no a los demás roles también es insuficiente.

[T1.5] Proporcionar contenidos avanzados específicos para el rol (herramientas, pilas de tecnología, ranking de errores de programación). La capacitación en seguridad del software va más allá de crear conciencia y de permitirles a los participantes incorporar prácticas de seguridad en su trabajo: está adaptada a los roles de estos, quienes obtienen información sobre herramientas, pilas de tecnologías y tipos de errores de programación que les son más relevantes. Una organización puede ofrecer, por ejemplo, una capacitación específica a cuatro perfiles distintos: una para arquitectos, una para desarrolladores Java, una para desarrolladores .NET y una cuarta para testers. Generalmente también se observa, entre los contenidos, la formación en herramientas específicas. No olvidar que la capacitación será de utilidad para muchos roles diferentes dentro de una organización, incluyendo QA, gestión de productos, ejecutivos y otros.

[T1.6] Crear y utilizar material específico sobre la historia de la empresa. Con el fin de generar un cambio fuerte y duradero en el comportamiento, la capacitación incluye material específico sobre la historia de la empresa. Cuando los participantes se pueden ver a sí mismos enfrentando un problema, es más probable que comprendan de qué manera el material es relevante para su trabajo y que sepan cuándo y cómo aplicar lo que han aprendido. Una forma de hacer esto es utilizar, como ejemplos en los contenidos de la capacitación, ataques ocurridos contra la empresa que sean dignos de atención. Se debe ser cauto cuando la capacitación trata sobre plataformas no utilizadas por los desarrolladores (a los desarrolladores para ambientes Windows no les interesan los viejos problemas de Unix) o contiene ejemplos de problemas que solo son relevantes en lenguajes que ya no son usados frecuentemente (no es necesario que los desarrolladores Java entiendan sobre los desbordamientos de memoria del C). Las historias tomadas de la propia empresa pueden ayudar a conducir la capacitación en la dirección correcta solo si las historias siguen siendo relevantes.

[T1.7] Proporcionar capacitación individual a demanda. La organización disminuye la carga de participantes y reduce los costos de capacitación al ofrecerla bajo demanda. La capacitación asistida por computadora es la opción más obvia y se la puede mantener actualizada a través de un modelo de suscripción. Estos cursos deben ser atractivos y relevantes para alcanzar el propósito deseado. En el caso de los desarrolladores también es posible proporcionar la capacitación de forma directa mediante los entornos de desarrollo integrado (IDE) adecuados, en el momento en que sea necesario. Recuerde, que en algunos casos, la capacitación con la asistencia de un instructor resulta más adecuada para el desarrollo de una nueva competencia (tal como la revisión de código).

dos

T Nivel 2: Crear un satélite de seguridad del software. El GSS debe construir y mejorar un satélite a través de actividades sociales, incluidas en ellas la capacitación y los eventos relacionados. El GSS y los gerentes deben garantizar que los nuevos empleados estén expuestos a la cultura de seguridad de la empresa durante las actividades de inducción.

[T2.5] Mejorar el satélite a través de capacitación y eventos. El GSS refuerza su red social mediante la celebración de eventos especiales destinados al satélite, en los que se lo capacita sobre temas avanzados o se presenta a oradores invitados. Puede ser un gesto, en estas ocasiones, ofrecer pizza y cerveza. Una videoconferencia no está comprendida en esta actividad, ya que ésta se orienta tanto a establecer lazos de camaradería como a compartir conocimiento o mejorar la eficiencia de la organización. Nada sustituye las reuniones presenciales, aun cuando solo sucedan una o dos veces al año.

[T2.6] Incluir recursos de seguridad en la etapa de inducción. El proceso que se sigue al incorporar empleados nuevos en la organización requiere de un módulo sobre la seguridad del software. El proceso genérico de inducción abarca cuestiones tales como que el empleado elija una buena contraseña y se asegure de que nadie se cuele detrás suyo al ingresar al edificio; puede ser mejorado aún más para que cubra temas como codificación segura, el SSDL y recursos de seguridad interna. El objetivo es garantizar que los empleados nuevos mejoren la cultura de seguridad. En las organizaciones de ingeniería, la rotación generalmente es alta. De todas formas, si bien resulta útil crear un módulo genérico dentro de la inducción, esto no reemplaza la realización oportuna de un curso introductorio más completo sobre seguridad del software.

[T2.7] Identificar el satélite a través de la capacitación. El satélite comienza como un conjunto de personas, diseminadas a lo largo de toda la organización, que muestran un nivel de interés o cierta competencia en seguridad superior a la media. La identificación de este grupo es un paso adelante en la creación de una red social que acelere la adopción de la seguridad en el desarrollo del software. Una forma de comenzar es realizar un seguimiento de las personas que se destacan durante los cursos de capacitación (véase [SM2.3Crear o hacer crecer un satélite]). En general, un ejército de voluntarios puede ser más fácil de conducir que uno reclutado.

tres

T Nivel 3: Fomentar el reconocimiento de las competencias y el progreso de la carrera profesional. La gerencia y el GSS deben asegurarse de que todos los miembros del personal reciban el reconocimiento apropiado por su progreso a lo largo de la capacitación, y también de elevar la moral. Los gerentes, los dueños de aplicaciones y el GSS deben capacitar a los proveedores y empleados subcontratados como una forma de propagar la cultura de la seguridad. Los gerentes y el GSS, además, deben continuar reforzando el impulso del satélite mediante la promoción en el exterior de la cultura de la seguridad. El GSS en particular debe estar disponible, al menos periódicamente, para aquellos que buscan orientación sobre la seguridad del software. Por su parte, los gerentes deben garantizar que todos los miembros del personal reciben capacitación por lo menos una vez al año.

[T3.1] Premiar el progreso logrado en la capacitación (certificación o RRHH). El conocimiento es una recompensa en sí mismo, pero premiar el progreso obtenido durante la capacitación en seguridad aporta otros beneficios. Los desarrolladores y los *testers* ven el realizar la capacitación en seguridad como una ventaja para su carrera. El sistema de recompensas puede ser formal, y dar lugar a una certificación o una calificación oficial en el sistema de RRHH, o informal, y hacer uso de motivadores tales como cartas de felicitación dirigidas al satélite y enviadas antes de la revisión anual. La participación de un departamento de capacitación o de Recursos Humanos puede, sí, volver más evidente el impacto de formarse en seguridad en la carrera profesional, pero el GSS debe continuar controlando el conocimiento sobre seguridad que se tiene dentro de la empresa y no ceder el control total o su supervisión.

[T3.2] Capacitar a los proveedores o empleados subcontratados. La organización proporciona capacitación en seguridad a proveedores y empleados subcontratados. Dedicar tiempo y esfuerzo ayudando a los proveedores a obtener correctos niveles de seguridad es más fácil que tratar de averiguar, más tarde, en qué se equivocaron. En el mejor de los casos, los empleados subcontratados reciben la misma capacitación que se brinda a los empleados. Capacitar a los empleados subcontratados individualmente antes que a todas las empresas subcontratadas es un camino que se toma generalmente y una forma razonable de comenzar. Por supuesto, es

importante capacitar a todos aquellos que trabajan en el desarrollo del software, independientemente de su modalidad laboral.

[T3.3] Organizar eventos externos relacionados con la seguridad del software. La organización destaca su cultura de seguridad como un elemento diferenciador mediante la celebración de eventos externos. BlueHat de Microsoft es un acontecimiento de este tipo, así como el Congreso de Seguridad de Intel. Los empleados se benefician al conocer las perspectivas externas, y la organización como un todo se beneficia al mostrar su credibilidad en materia de seguridad (véase [SM3.2 Ejecutar un programa de marketing externo]).

[T3.4] Exigir un curso de actualización anual. Se requiere que todos los involucrados en la producción de software participen anualmente de un curso de actualización acerca de la seguridad del software. Este curso mantiene actualizado al personal en materia de seguridad y garantiza que la organización no pierda su foco debido a la rotación. El GSS puede usar la mitad de un día para dar una actualización sobre el panorama de seguridad y explicar los cambios en las políticas y normas. Un curso de actualización se puede realizar como parte del día de la seguridad para toda la empresa o conjuntamente con una conferencia interna sobre seguridad.

[T3.5] Establecer los horarios de la oficina del GSS. El GSS ofrece ayuda a todos los interesados durante horarios especiales anunciados o regularmente programados. Al actuar como un recurso informal para las personas que quieren resolver problemas de seguridad, el GSS aprovecha estos momentos de enseñanza y motiva el aprendizaje. Los horarios de esta oficina se podrían establecer en una vez por semana durante la tarde, en la oficina de un miembro experimentado del GSS. También es una posibilidad determinar horarios de oficina móviles, con visitas a grupos particulares de producto o de aplicación, programadas por petición.

INTELIGENCIA: Modelos de Ataque (AM)

La meta general de la práctica Modelos de Ataque es la creación de conocimiento personalizado acerca de los ataques relevantes para la organización. Este conocimiento debe guiar las decisiones sobre la codificación y los controles.

uno

AM Nivel 1: Crear una base de conocimiento sobre activos de datos y ataques. El GSS debe identificar a los potenciales atacantes y documentar tanto los ataques que causen mayor preocupación en la organización como cualquier ataque importante que haya ocurrido. También debe comunicar la información sobre el atacante a todas las partes interesadas. La empresa debe crear un esquema de clasificación de datos que el GSS utilice para inventariar y priorizar las aplicaciones.

[AM1.1] Construir y mantener una lista de los N principales posibles ataques. El GSS ayuda a la organización a comprender conceptos básicos sobre los ataques manteniendo una lista de los más importantes para la empresa y la utiliza para guiar el cambio. Esta lista combina aportes de múltiples fuentes: ataques observados, foros de hackers, tendencias de la industria, etc. La lista no tiene por qué ser actualizada frecuentemente y los ataques pueden estar ordenados a grandes trazos. Por ejemplo, el GSS podría realizar dos veces al año una reunión del tipo "torbellino de ideas" para crear listas de aquellos ataques que la organización debería estar preparada para combatir "ahora", "pronto" y "algún día". En algunos casos, se utiliza información del modelo de ataque en un enfoque basado en listas para el análisis de la arquitectura, lo que ayuda a concentrar el estudio, como en el caso de STRIDE.

[AM1.2] Crear un esquema de clasificación e inventario de datos. La organización determina un esquema de clasificación de datos y lo utiliza para hacer un inventario de su software de acuerdo con los tipos de datos que el software maneje. Esto permite priorizar las aplicaciones en base a la clasificación de sus datos. Existen muchos esquemas de clasificación: uno consiste en focalizarse en la IIP. Dependiendo del esquema y el software involucrado, puede resultar más útil clasificar primero los repositorios de datos y luego derivar esas clasificaciones a las aplicaciones en función de los repositorios que utilicen. Hay también otras maneras de abordar el problema. Por ejemplo, los datos se pueden clasificar de acuerdo a criterios de protección de la propiedad intelectual, del impacto que provocaría su divulgación, de su exposición a ataques, de su importancia para marcos regulatorios específicos (tal como SOX) o de fronteras geográficas.

[AM1.3] Identificar a potenciales atacantes. El GSS identifica a potenciales atacantes con el fin de comprender sus motivaciones y capacidades. El resultado de este ejercicio puede ser un conjunto de perfiles de atacantes, en el que estén incluidos bocetos genéricos para amplias categorías de atacantes y descripciones más detalladas para individualidades notables. En algunos casos, se puede contratar a un proveedor externo que proporcione esta información. Casi siempre es más útil la información específica y contextual sobre un atacante que la información genérica copiada de la lista de algún otro.

[AM1.4] Recopilar y publicar historias de ataques. Con el fin de maximizar el beneficio de lecciones que no siempre son poco costosas, el GSS recopila y publica historias sobre ataques contra la organización. Con el tiempo, esta colección ayuda a la organización a comprender su historia. Tanto los ataques exitosos como los que no lo fueron pueden resultar notables. Discutir la información histórica de los ataques contra la organización tiene el efecto de fundamentar la seguridad del software en la realidad de una empresa. Esto se muestra particularmente útil para ser usado en las actividades de capacitación, con el fin de contrarrestar un abordaje genérico que se enfoque solo en listas de "los diez principales" o en ataques a plataformas irrelevantes y obsoletas. Ocultar información acerca de los ataques a aquellas personas que construyen nuevos sistemas no favorece la posibilidad de cosechar efectos positivos de un suceso negativo.

[AM1.5] Reunir inteligencia sobre ataques. El GSS se mantiene por delante de la curva de aprendizaje sobre los nuevos tipos de ataques y vulnerabilidades. La información puede provenir de la asistencia a conferencias y talleres, del seguimiento de foros de atacantes y de la lectura de publicaciones relevantes, listas de correo y blogs. Enorgullecer a Sun Tzu conociendo al enemigo: debe relacionarse con aquellos investigadores de seguridad que le pueden causar problemas. En muchos casos, una suscripción a un servicio comercial

proporciona una vía razonable para recolectar inteligencia sobre aquello en lo que se basa un ataque. Independientemente de su origen, la información sobre ataques debe poder ser procesable y de utilidad para los constructores de software y *testers*.

[AM1.6] Crear un foro interno donde discutir los ataques. La organización cuenta con un foro interno donde el GSS, el satélite y otros discuten los ataques. El foro sirve para comunicar la perspectiva del atacante. El GSS podría mantener una lista de correo interna donde los suscriptores compartan la información más reciente sobre incidentes conocidos públicamente. Puede resultar especialmente útil realizar un análisis detallado de los ataques y los exploits que son relevantes para una empresa, sobre todo si estimula la discusión sobre posibles mitigaciones durante el desarrollo. El mero reenvío de información proveniente de listas de distribución públicas no ofrece los mismos beneficios que una discusión activa. Vigilancia significa "nunca estar demasiado cómodo". (Véase [SR1.2 Crear un portal de seguridad]).

dos

AM Nivel 2: Divulgar información acerca de atacantes y ataques relevantes. El GSS debe recolectar inteligencia sobre ataques y difundir ese conocimiento, para que sea incluido tanto en los patrones de ataque de alto nivel como en los casos de abuso de bajo nivel. Los patrones de ataque deben incluir información sobre tecnologías específicas relevantes para la organización.

[AM2.1] Construir patrones de ataque y casos de abuso ligados a potenciales atacantes. El GSS se prepara para las pruebas de seguridad y el análisis de la arquitectura mediante la construcción de patrones de ataque y casos de abuso ligados a potenciales atacantes. Para ser de utilidad, estos recursos no se tienen que construir desde cero para cada aplicación; en su lugar, podría haber conjuntos estándares para aplicaciones con perfiles similares. El GSS agregará los patrones y casos basándose en las historias de ataque. Por ejemplo, una historia sobre un ataque contra permisos pobremente administrados puede dar lugar a un patrón de ataque sobre permisos que determina un nuevo tipo de pruebas. Otro ejemplo: si una empresa sigue las pistas de un fraude y de los costos monetarios asociados a los ataques particulares, esta información se puede utilizar para guiar el proceso de construcción de patrones de ataque y casos de abuso.

[AM2.2] Crear patrones de ataque para tecnologías específicas. El GSS crea patrones de ataque para tecnologías específicas, con el fin de capturar el conocimiento sobre ataques que se construyen sobre esas tecnologías en particular. Por ejemplo, si el software web de la organización se basa en las capacidades avanzadas del navegador, el GSS podría catalogar dichas capacidades en los navegadores más populares y analizar cómo se podrían explotar sus vulnerabilidades. Pueden ser de utilidad los patrones de ataque directamente relacionados con la frontera del conocimiento de seguridad (en la actualidad, seguridad en dispositivos móviles y seguridad en la nube). La mera reiteración de directrices generales –por ejemplo, "garantizar que los datos en tránsito estén protegidos"– con el agregado al final de "en las aplicaciones móviles" no constituye patrones de ataque para tecnologías específicas.

tres

AM Nivel 3: Investigar y mitigar los nuevos patrones de ataque. El GSS debe dirigir la investigación sobre ataques contra el software empresarial para llevar la delantera respecto de la actividad del atacante. También debe proporcionar el conocimiento y la automatización a auditores y testers para garantizar que sus actividades reflejen tanto ataques verdaderos perpetrados contra el software de la organización como otros potenciales ataques.

[AM3.1] Tener un equipo científico que desarrolla nuevos métodos de ataque. El GSS cuenta con un equipo científico que trabaja para identificar y desactivar nuevas clases de ataques antes de que los atacantes reales siquiera sepan que existen. No es un equipo de pruebas de penetración que está a la búsqueda de nuevas instancias de los tipos conocidos de debilidades: es un grupo de investigación que innova en nuevos tipos de ataques. Un equipo científico puede incluir conocidos investigadores de seguridad que publiquen sus hallazgos en conferencias como la Def Con.

[AM3.2] Crear y emplear métodos automatizados para hacer lo que harán los atacantes. El GSS pone a disposición de *testers* y auditores métodos automatizados para hacer lo que los atacantes harán. Por ejemplo, un nuevo método

de ataque identificado por el equipo científico podría requerir una nueva herramienta; el GSS empaquetaría entonces ese nuevo instrumento y lo distribuiría entre los testers. La idea aquí es incrementar las capacidades de ataque más allá de lo que incluyen por defecto las herramientas comerciales y otras ofertas típicas, y luego empaquetar esa información o instrumento para que otros los utilicen. Adaptar las herramientas a las pilas de tecnologías y a los potenciales atacantes particulares de una empresa es una idea muy buena.

INTELIGENCIA: Características de Seguridad y Diseño (SFD)

La meta general de la práctica Características de Seguridad y Diseño es la creación de conocimiento personalizado sobre las características, los marcos de referencia y los patrones de seguridad. Este conocimiento debe guiar las decisiones acerca de la arquitectura y los componentes.

uno

SFD Nivel 1: Publicar las características de seguridad y la arquitectura. El GSS les debe proporcionar orientación sobre las características de seguridad del software a los arquitectos y los desarrolladores, y colaborar de forma directa con los grupos de arquitectos.

[SFD1.1] Construir y publicar características de seguridad. Algunos problemas se resuelven mejor sólo una vez. En lugar de hacer que cada equipo de proyecto implemente sus propias características de seguridad del software (autenticación, gestión de roles, gestión de claves, auditoría/logs, criptografía, protocolos, etc.), el GSS proporciona una orientación proactiva mediante la construcción y publicación de características de seguridad para que todos los grupos los utilicen. Los equipos de proyecto se benefician de las implementaciones que fueron aprobadas previamente por el GSS, y el GSS se beneficia al no tener que hacer varias veces un rastreo buscando los mismos tipos de errores sutiles que a menudo se introducen en las características de seguridad de un software. El GSS puede identificar una implementación que le agrade y promoverla como la solución aceptada.

[SFD1.2] Involucrar al GSS con la arquitectura. La seguridad es un tema de discusión habitual en lo que hace a la arquitectura del software de una organización. El grupo de arquitectura asume la responsabilidad por la seguridad del mismo modo que lo hace por el rendimiento, la disponibilidad o la escalabilidad del software. Una manera de evitar que la seguridad quede fuera de la discusión es hacer que un miembro del GSS asista a las reuniones habituales de arquitectura. En otros casos, la arquitectura empresarial puede ayudar al GSS a crear diseños seguros que se integren adecuadamente a las normas corporativas de diseño.

dos

SFD Nivel 2: Construir e identificar soluciones de seguridad. El GSS debe proporcionar marcos de referencia seguros por diseño (*secure by design*), y estar disponible y tener la capacidad de resolver problemas de diseño para los demás.

[SFD2.1] Construir marcos de referencia de *middleware* y bibliotecas comunes, seguros por diseño. El GSS toma un rol proactivo en el diseño de software mediante la construcción o recomendación de marcos de referencia de *middleware* o bibliotecas comunes seguros desde su diseño. Además de enseñar con el ejemplo, este *middleware* ayuda en el análisis de la arquitectura y la revisión de código, ya que los bloques constructivos hacen que sea más fácil detectar errores. Por ejemplo, el GSS puede modificar un marco de referencia web de uso muy difundido, como Spring, para que sea fácil cumplir con los requisitos de validación de entrada. Finalmente, el GSS puede adaptar las reglas de revisión de código específicamente para los componentes que ofrece (véase [CR3.1 Utilizar herramientas automatizadas con reglas a medida]). Cuando se adopta un marco de referencia de *middleware* (o cualquier otro software ampliamente utilizado), es importante realizar un examen cuidadoso de su seguridad antes de su publicación. Fomentar la adopción y la utilización de middleware inseguro no ayuda al estado de la seguridad del software. Las arquitecturas genéricas para seguridad del software de código abierto, incluida OWASP ESAPI, no se deben considerar seguras a priori.

[SFD2.2] Generar la capacidad en el GSS de resolver problemas difíciles de diseño. Cuando el GSS se involucra en el proceso de un proyecto nuevo de manera temprana, contribuye con la arquitectura nueva y resuelve problemas difíciles de diseño. El impacto negativo que tiene la seguridad sobre otras restricciones (tiempo de comercialización, precio, etc.) se reduce al mínimo. Si un arquitecto del GSS está involucrado en el diseño de un protocolo nuevo, puede analizar las implicancias de seguridad de los protocolos existentes e identificar los elementos que se deberían duplicar o evitar. Diseñar anticipadamente la seguridad es más eficiente que analizar un diseño existente y luego tener que refactorizarlo cuando se descubren defectos. Algunos problemas de diseño requerirán experiencias específicas por fuera del GSS.

tres

SFD Nivel 3: Reutilizar activamente características de seguridad y marcos de referencia seguros por diseño aprobados. El SSG debe proporcionar patrones de diseño maduros adicionales tomados del software y de las pilas de tecnología existentes. Los administradores deben asegurarse de que existe consenso formal en la organización acerca de las elecciones de diseño seguro. También deben exigir que se utilice, siempre que sea posible, características de seguridad y marcos de referencia definidos.

[SFD3.1] Formar un consejo o comité central de revisión para aprobar y mantener los patrones de diseño seguro. Un consejo o comité central de revisión formaliza el proceso para llegar a un consenso sobre el equilibrio entre las necesidades de diseño y las de seguridad. A diferencia del comité de arquitectura, este grupo se centra específicamente en proporcionar orientación en seguridad. El grupo también puede revisar periódicamente las normas de diseño publicadas (especialmente las relativas a criptografía) para garantizar que las decisiones de diseño no se vuelvan obsoletas o desactualizadas.

[SFD3.2] Exigir la utilización de características de seguridad y marcos de referencia ya aprobados. Los encargados de las implementaciones deben tomar sus características de seguridad y marcos de referencia de una lista aprobada. Existen dos beneficios: los desarrolladores programadores no pierden el tiempo reinventando funcionalidades/características ya existentes, y los equipos de revisión no tienen que lidiar con la búsqueda de los mismos viejos defectos en proyectos nuevos. En concreto, cuando se utiliza componentes aprobados en un proyecto nuevo, se vuelven más fáciles el análisis de la arquitectura y la revisión de código (véase [AA1.1 Llevar a cabo la revisión de las características de seguridad]). La reutilización es una de las principales ventajas de una arquitectura de software consistente.

[SFD3.3] Encontrar y publicar patrones maduros de diseño de la organización. El GSS fomenta la reutilización del diseño centralizado mediante la búsqueda y publicación de patrones maduros de diseño de y en la organización. Una sección del sitio web del GSS podría promover los elementos positivos identificados durante el análisis de la arquitectura de modo que se propaguen las buenas ideas. Este proceso debe estar formalizado; no es suficiente que un grupo ad hoc se entere de manera accidental. En algunos casos, la participación de un equipo central de arquitectura o de tecnología facilita y mejora esta actividad.

INTELIGENCIA: Normas y requisitos (SR)

La meta general de la práctica Normas y Requisitos es la creación de lineamientos prescriptivos para todos los interesados. Los administradores y el GSS deben documentar las opciones de seguridad del software y transmitir ese material a todos los involucrados en el SSDL, incluidos los terceros interesados.

uno

SR Nivel 1: Proporcionar normas y requisitos de seguridad de fácil acceso. El GSS debe proporcionar conocimientos básicos que incluyan al menos: normas de seguridad, normas de codificación segura y requisitos de cumplimiento. Los administradores deben garantizar que la información de seguridad del software se mantenga actualizada y disponible para todos.

[SR1.1] Crear normas de seguridad. La seguridad del software requiere de mucho más que solo las características de seguridad, aunque estas también sean parte del trabajo a hacer. El GSS satisface la demanda de la organización en lo que hace a sus lineamientos de seguridad mediante la creación de normas que explican la manera aceptada de adherir a las políticas y llevar a cabo operaciones específicas centradas en la seguridad. Una norma puede describir cómo llevar a cabo la autenticación si utiliza J2EE o cómo determinar la autenticidad de una actualización de software (en [SFD1.1 Construir y publicar las características de seguridad] puede encontrar un caso en el que el GSS proporciona una implementación de referencia de una norma de seguridad). Las normas pueden desplegarse de maneras variadas. En algunos casos, las normas y las directrices pueden automatizarse en los entornos de desarrollo (por ejemplo, estar aplicadas dentro de un IDE). En otros casos, las directrices pueden estar explícitamente vinculadas a ejemplos de código para mejorar su observancia y hacerlas más relevantes.

[SR1.2] Crear un portal de seguridad. La organización cuenta con una ubicación central donde colocar la información sobre seguridad del software. Normalmente se trata de un sitio web interno mantenido por el GSS. Las personas recurren al portal para obtener las últimas y mejores normas y requisitos de seguridad, así como otros recursos provistos por el GSS. Una wiki interactiva resulta mejor que un portal estático con documentos de orientación que rara vez son actualizados. Las organizaciones pueden complementar estos materiales con listas de correo y reuniones presenciales.

[SR1.3] Traducir las restricciones de cumplimiento en requisitos. Las restricciones de cumplimiento se traducen en requisitos de software para los proyectos individuales. Esto es una pieza clave en la estrategia de cumplimiento de la organización: a través de la representación explícita de las restricciones de cumplimiento en requisitos, se demuestra que el cumplimiento puede ser una tarea gestionable. Por ejemplo, si la organización habitualmente construye software que procesa transacciones de tarjetas de crédito, el cumplimiento de PCI DSS podría desempeñar un rol en el SSDL durante la fase de definición de requisitos. En otros casos, las normas tecnológicas definidas por razones de interoperabilidad internacional pueden incluir orientación de seguridad. La representación de estas normas como requisitos contribuye a la trazabilidad y visibilidad, en el caso de auditorías.

[SR1.4] Utilizar normas de codificación segura. Las normas de codificación ayudan a los desarrolladores a evitar los errores de programación más obvios y proporcionan normas básicas para la revisión de código. Son, necesariamente, específicas para un lenguaje de programación, y pueden utilizar marcos de referencia de uso generalizado y bibliotecas. Si la organización ya cuenta con normas de codificación para otros propósitos, las de codificación segura se deben construir sobre ellas. Un conjunto claro de normas de codificación segura es una buena manera de guiar la revisión de código tanto manual como automatizada, a la vez que refuerza la capacitación en seguridad con ejemplos relevantes.

dos

SR Level 2: Comunicar las normas formalmente aprobadas, tanto de manera interna como a los proveedores. Los gerentes deben garantizar que se utilice un proceso formal para la creación de normas específicas para las pilas de tecnología. Los gerentes, el GSS y los dueños de producto deben garantizar la aplicación de los SLA utilizando el lenguaje contractual aprobado por el equipo de legales. El GSS debe asegurar que todo el software de código abierto está identificado dentro del código de la organización.

[SR2.2] Crear un comité de revisión de normas. La organización crea un comité de revisión de normas a fin de formalizar el proceso utilizado para elaborarlas y de garantizar que todos los interesados tengan la oportunidad

de sopesarlas. El comité de revisión podría operar mediante el nombramiento, para cada norma propuesta, de un promotor que le dé impulso. La responsabilidad de demostrar que la norma cumple con sus metas y de obtener la aprobación y aceptación por parte del comité de revisión recaería en el promotor. Los grupos de arquitectura o de riesgo empresarial algunas veces asumen la responsabilidad de crear y gestionar los comités de revisión de normas de la empresa.

[SR2.3] Crear normas para pilas de tecnología. La organización establece normas relativas a pilas de tecnología específicas. Para el GSS esto significa una menor carga de trabajo, debido a que no tendrá que buscar riesgos tecnológicos viejos en cada proyecto nuevo. Idealmente, la organización creará una configuración base segura para cada pila de tecnología, lo que reducirá aún más la cantidad de trabajo requerido para utilizarla de manera segura. Una pila puede incluir un sistema operativo, una base de datos, un servidor de aplicaciones y un ambiente de ejecución para un lenguaje administrado. La frontera de la seguridad es un buen lugar para progresar en ese sentido. Actualmente, tanto las pilas y plataformas de tecnologías móviles como las basadas en la nube son dos áreas donde la atención específica a la seguridad se ve retribuida.

[SR2.4] Identificar el código abierto. El primer paso hacia la gestión del riesgo introducido por el código abierto es identificar los componentes de este tipo en uso. No es raro descubrir versiones antiguas de componentes, con vulnerabilidades conocidas, o múltiples versiones del mismo componente. Una manera de abordar esta actividad es a través de herramientas automatizadas para la búsqueda de código abierto. Un proceso que sólo se basa en que los desarrolladores soliciten permiso no genera resultados satisfactorios. En el siguiente nivel de madurez, esta actividad está subsumida en una política de restricción del uso de código abierto.

[SR2.5] Crear un modelo de SLA. El GSS trabaja con el departamento de legales en la creación de un texto modelo de SLA estándar para utilizar en los contratos con los proveedores y proveedores subcontratados. El departamento de legales entiende que el texto modelo ayuda a evitar problemas de cumplimiento y privacidad. Según el acuerdo, los proveedores y proveedores subcontratados deben cumplir con las normas de seguridad del software de la compañía (véase [CP2.4 Incluir en todos los contratos con proveedores acuerdos de nivel de servicio de seguridad del software (SLA)].) El texto modelo puede solicitar al vendedor soluciones con control de seguridad del software tales como mediciones vBSIMM o puntajes BSIMM.

tres

SR Nivel 3: Exigir decisiones de gestión del riesgo para la utilización de código abierto. Los administradores y el GSS deben demostrar que cualquier código abierto utilizado en la organización está sujeto a los mismos procesos de gestión del riesgo que el código creado internamente, y garantizar que todas las normas aplicables se comunican a los proveedores.

[SR3.1] Control del riesgo del código abierto. La organización tiene control sobre la exposición a las vulnerabilidades concomitante a la utilización de componentes de código abierto. El uso de código abierto se puede restringir a proyectos predefinidos; también, a versiones que hayan pasado por un proceso de inspección de seguridad del GSS, con vulnerabilidades inaceptables remediadas y disponibles solo a través de repositorios internos. Las cuestiones legales a menudo demandan controles adicionales del código abierto, debido principalmente al problema de "licencia viral" asociada al código liberado bajo la GPL. Conseguir que el área de legales comprenda los riesgos de seguridad puede ayudar a que la organización comience a practicar una higiene adecuada del código abierto.

[SR3.2] Comunicar las normas a los proveedores. El GSS trabaja con los proveedores para educarlos y promover las normas de seguridad de la organización. Una relación sana con un vendedor no se puede garantizar sólo apelando a un lenguaje contractual. El GSS se relaciona con los proveedores, discute las prácticas de seguridad de estos y les explica en términos concretos (en lugar de términos legales) lo que la organización espera de ellos. Cada vez que un proveedor adopta las normas de seguridad de la organización, es una clara victoria. Cuando el SSDL de la empresa está disponible públicamente, la comunicación de las expectativas en materia de seguridad del software resulta más sencilla. Del mismo modo, el intercambio de prácticas y medidas internas (incluida una medición BSIMM) puede clarificar muy bien cuáles son las expectativas.

PUNTOS DE CONTACTO CON EL SSDL : Análisis de la Arquitectura (AA)

La meta general de la práctica Análisis de la Arquitectura es el control de calidad. Quienes llevan a cabo el análisis de la arquitectura deben garantizar la detección y corrección de fallas de seguridad. Los arquitectos de software deben hacer cumplir la adhesión a las normas y la reutilización de las características de seguridad aprobadas.

uno

AA Nivel 1: Llevar a cabo revisiones del análisis de la arquitectura (AA) guiadas por el riesgo, dirigidas por el GSS. La organización debe proporcionar una clasificación ligera del riesgo del software. El GSS debe comenzar dirigiendo los esfuerzos del análisis de la arquitectura, especialmente en aplicaciones de alto riesgo, como una manera de construir capacidad interna y de demostrar el valor de la acción a nivel del diseño.

[AA1.1] Llevar a cabo la revisión de las características de seguridad. Para comenzar con el análisis de la arquitectura, el proceso se debe centrar en la revisión de las características de seguridad. Los revisores concientizados en seguridad del software primero identifican las características de seguridad de una aplicación (autenticación, control de acceso, uso de criptografía, etc.), para luego estudiar el diseño en busca de los problemas que podrían causarse si dichas características fallaran en su propósito o resultasen insuficientes. Por ejemplo, en este tipo de revisión se podría identificar que un sistema que fue objeto de ataques de escalamiento de privilegios falló debido al quiebre del control de acceso o del sistema que almacena los valores *hash* de las contraseñas. En los niveles más altos de madurez, esta actividad queda eclipsada por un enfoque más completo del análisis de la arquitectura que no está centrado en las características. En algunos casos, el uso de componentes seguros por diseño de la empresa puede agilizar este proceso.

[AA1.2] Llevar a cabo una revisión del diseño en aplicaciones de alto riesgo. La organización aprende sobre los beneficios del análisis de la arquitectura al ver resultados reales en aplicaciones de alto riesgo y alto perfil. Los revisores deben tener alguna experiencia en la realización de análisis de arquitecturas y en "romper" la arquitectura que están evaluando. Si el GSS aún no está preparado para llevar a cabo un análisis en profundidad de la arquitectura, debe recurrir a consultores para realizar este trabajo. Aquí se pueden utilizar paradigmas de revisión ad hoc que dependerán en gran medida de la experiencia, aunque a la larga no escalan.

[AA1.3] Tener al GSS como líder de los esfuerzos de revisión del diseño. El GSS toma un rol protagónico en la realización del análisis de la arquitectura con el fin de comenzar a construir en la organización capacidades para descubrir defectos de diseño. "Romper" la arquitectura es casi un arte, por lo que requiere que el GSS sea competente en esto antes de poder delegar el trabajo a los arquitectos, y la competencia requiere práctica. El GSS no puede tener éxito por sí mismo: probablemente necesite ayuda de los arquitectos o desarrolladores para poder entender el diseño. Con un diseño claro en mano, el GSS puede realizar el análisis con una mínima interacción con el equipo del proyecto. En los niveles más altos de madurez, la responsabilidad de liderar los esfuerzos de revisión se desplaza hacia los arquitectos de software. Los enfoques del análisis de la arquitectura (y del modelado de amenazas) evolucionan con el tiempo. No espere establecer un proceso y utilizarlo para siempre.

[AA1.4] Utilizar un cuestionario sobre riesgos para clasificar las aplicaciones. Para facilitar el análisis de la arquitectura y otros procesos, el GSS utiliza un cuestionario sobre riesgos para recopilar información básica de cada aplicación y así poder determinar una clasificación de riesgos y un esquema de asignación de prioridades. La lista de preguntas puede incluir: "¿En qué lenguajes de programación está escrita la aplicación?", "¿quién utiliza la aplicación?" y "¿la aplicación maneja IIP?". Un miembro calificado del equipo de aplicación debe completar el cuestionario; este debe ser lo suficientemente corto como para poder ser completado en cuestión de horas. El GSS puede utilizar las respuestas para clasificar una aplicación como de alto, medio o bajo riesgo. Debido a que un cuestionario de riesgos se puede responder de manera ligera, es importante que tengan lugar algunas inspecciones in situ para corroborar su validez y exactitud. Una excesiva dependencia en el autoexamen o la automatización pueden hacer que esta actividad resulte de poca ayuda.

dos

AA Nivel 2: Divulgar la utilización de un proceso para el análisis de la arquitectura (AA) documentado. El GSS debe facilitar la utilización del análisis de la arquitectura en toda la organización, brindando su disponibilidad como

recurso o como mentor. El GSS debe definir un proceso de análisis de la arquitectura basado en un lenguaje común de descripción de la arquitectura y en modelos de ataque estándares.

[AA2.1] Definir y utilizar el proceso de análisis de la arquitectura. El GSS define y documenta un proceso para llevar a cabo el análisis de la arquitectura y lo aplica en las revisiones de diseño que conduce. El proceso incluye un enfoque estandarizado para pensar acerca de los ataques y las propiedades de seguridad, y se define con el suficiente rigor como para que se les pueda enseñar a realizarlo a quienes no pertenecen al GSS. Se debe prestar especial atención a la documentación, tanto la de la arquitectura bajo revisión como la de los defectos de seguridad descubiertos. El "conocimiento tribal" (*tribal knowledge*) no cuenta como un proceso definido. STRIDE de Microsoft y ARA de Cigital son ejemplos de tales procesos. Tener en cuenta que incluso estas dos metodologías para el análisis de la arquitectura evolucionaron mucho a lo largo del tiempo. Se debe garantizar el acceso a fuentes actualizadas con información de análisis de la arquitectura, dado que muchas de las primeras publicaciones de las metodologías de este tipo son obsoletas y no se aplican más.

[AA2.2] Normalizar las descripciones arquitectónicas (incluido el flujo de datos). La organización utiliza un formato acordado para la descripción de la arquitectura, incluido un medio para representar el flujo de datos. Este formato, combinado con un proceso de análisis de la arquitectura, hace que dicho análisis resulte manejable incluso para quienes no son expertos en seguridad. Una descripción normalizada de la arquitectura se puede mejorar a fin de ofrecer una imagen explícita de los activos de la información que requieren protección. Resulta especialmente útil emplear íconos normalizados que sean usados de manera consistente en diagramas UML, plantillas de Visio y bosquejos.

[AA2.3] Disponer del GSS como recurso o mentor del análisis de la arquitectura. Con el fin de construir una capacidad de análisis de la arquitectura que esté por fuera del GSS, este se anuncia a sí mismo como un recurso o mentor para los equipos que pidan ayuda al realizar sus propias revisiones de diseño, y busca proactivamente proyectos en los cuales involucrarse. El GSS responderá a preguntas acerca de análisis de la arquitectura durante el horario de trabajo y, en algunos casos, puede asignar a alguien para que se siente a trabajar codo a codo con el arquitecto durante el análisis. En el caso de aplicaciones o productos de alto riesgo, el GSS desempeña un rol de mentor más activo.

tres

AA Nivel 3: Construir capacidades de revisión y de recuperación dentro del grupo de arquitectura. Los arquitectos de software deben conducir los esfuerzos de análisis a lo largo de la organización y utilizar los resultados de los análisis para actualizar y crear patrones de arquitectura estándares que sean seguros.

[AA3.1] Contar con arquitectos de software que lideren los esfuerzos de revisión del diseño. Los arquitectos de software de la organización lideran el proceso de análisis de la arquitectura la mayor parte del tiempo. El GSS aún podría contribuir con dicho análisis, en carácter consultivo o bajo circunstancias especiales. Esta actividad requiere de un proceso de análisis de la arquitectura bien comprendido y bien documentado. Incluso en ese caso, es muy difícil conseguir consistencia, porque "romper" arquitecturas requiere de mucha experiencia.

[AA3.2] Traducir los resultados del análisis a patrones de arquitectura estándares. Las fallas identificadas durante los análisis de la arquitectura se remiten al comité de diseño de seguridad para que en el futuro, al crear patrones de diseño mejorados, se puedan evitar errores similares (véase [SFD3.1 Formar un consejo o comité central de revisión para aprobar y mantener los patrones de diseño seguros]) Los patrones de diseño de seguridad pueden interactuar de maneras sorprendentes para quebrar la seguridad. El proceso de análisis de la arquitectura se debería aplicar incluso cuando los patrones de diseño examinados sean de uso habitual.

PUNTOS DE CONTACTO CON EL SSDL: Revisión de Código (CR)

La meta general de la práctica Revisión de Código es el control de calidad. Quienes llevan a cabo la revisión de código deben garantizar la detección y corrección de errores de seguridad. El GSS debe hacer cumplir la adhesión a las normas y que se reutilicen las características de seguridad aprobadas.

uno

CR Nivel 1: Revisar el código en forma manual y automatizada, con informes centralizados. El GSS debe ponerse a disposición de los demás para elevar el nivel de concientización en la revisión de código y demandar su realización. El GSS debe llevar a cabo revisiones de código de aplicaciones de alto riesgo, siempre que pueda participar en el proceso, y debe utilizar los conocimientos adquiridos para informar a la organización de los tipos de errores de programación en el software que fueran descubiertos. La gerencia debe hacer obligatoria la revisión de código para todos los proyectos de software. Por otra parte, el GSS debe imponer la utilización de herramientas centralizadas para la generación de informes, a fin de capturar conocimiento sobre errores de programación recurrentes e incluir esta información en la estrategia y la capacitación.

[CR1.1] Crear una lista de los N principales errores de programación (preferir datos reales). El GSS mantiene una lista de los tipos de errores de programación más importantes que se deben eliminar del código de la organización, y la utiliza para conducir el cambio. La lista ayuda a centrar la atención de la organización en aquellos errores de programación que más importan. Se puede seleccionar una lista genérica de fuentes públicas, pero es mucho más útil si el listado es específico a la organización y se lo construye a partir de datos reales recopilados de la revisión de código, pruebas e incidentes reales. El GSS puede actualizar periódicamente la lista y publicar un informe de los "más buscados". (Para conocer otra forma de usar la lista, vea [T1.6 Crear y utilizar material específico sobre la historia de la empresa]). Algunas empresas utilizan múltiples herramientas y datos reales de la base de código para crear listas de los N principales, sin limitarse a un determinado servicio o herramienta. Una dificultad potencial con una lista de los N principales es el problema conocido como "buscar las llaves solo bajo la luz de la calle". Por ejemplo, la lista OWASP *Top Ten* rara vez refleja las prioridades de los errores de programación de una organización. Ordenar, simplemente, los datos de errores de programación del día por el número de ocurrencias no produce una lista satisfactoria de los N principales, ya que estos datos cambian muy a menudo.

[CR1.2] Hacer que el GSS lleve a cabo revisiones ad hoc. En el caso de aplicaciones de alto riesgo, el GSS lleva a cabo una revisión ad hoc de código de manera oportuna. Por ejemplo, el GSS podría proseguir la revisión de diseño de estas aplicaciones con una revisión de código. Reemplazar una orientación ad hoc con un enfoque sistemático en los niveles más altos de madurez. La revisión del GSS puede incluir el uso de herramientas y servicios específicos, o puede ser manual.

[CR1.4] Utilizar herramientas automatizadas junto con la revisión manual. Se incorpora el análisis estático en el proceso de revisión de código para hacer más eficiente y consistente dicha revisión. La automatización no reemplaza el juicio humano, pero sí brinda mayor definición al proceso de revisión y experiencia en seguridad a los revisores que no son expertos en el tema. Una empresa puede utilizar un proveedor externo de servicios como parte de un proceso formal de revisión de código para la seguridad del software. Este servicio debería estar explícitamente conectado a un SSDL más grande aplicado durante el desarrollo de software y no sólo para "tildar la casilla de seguridad" en el camino hacia el despliegue.

[CR1.5] Hacer obligatoria la revisión de código para todos los proyectos. La revisión de código es una "puerta" de cruce obligatorio para el lanzamiento en todos los proyectos bajo el ámbito del GSS. La falta de revisión de código o resultados inaceptables detendrán su lanzamiento. Si bien todos los proyectos deben someterse a la revisión de código, el proceso de revisión puede ser diferente para distintos tipos de proyectos. La revisión de proyectos de bajo riesgo puede depender más de la automatización, y la revisión de proyectos de alto riesgo, no tener límite en la cantidad de tiempo empleado por los revisores. En la mayoría de los casos, una "puerta" de revisión de código que tenga un estándar mínimo pero aceptable fuerza a que los proyectos que no lo pasen deban ser corregidos y evaluados nuevamente antes de su lanzamiento.

[CR1.6] Utilizar informes centralizados para cerrar el ciclo de conocimiento y conducir la capacitación. El seguimiento de los errores de programación encontrados durante la revisión de código se realiza en un repositorio centralizado. Este repositorio permite elaborar informes resumidos e informes de tendencia para la organización. El GSS puede utilizar los informes para demostrar el progreso y guiar los contenidos de la capacitación (véase [SM2.5 Identificar métricas y utilizarlas para guiar los presupuestos]). La información surgida de la revisión de código se puede incorporar en un panel de control gerencial (*CSO-level*) alimentado también desde otras partes de la organización de seguridad. Del mismo modo, también puede alimentar a un sistema de seguimiento de proyectos que abarque todo el desarrollo y que incorpore una cantidad de fuentes diversas de seguridad del software (por ejemplo, pruebas de penetración, pruebas de seguridad, pruebas de caja negra, pruebas de caja blanca, etc.). No olvide que los errores de programación individuales constituyen excelentes ejemplos para la capacitación.

dos

CR Nivel 2: Hacer cumplir las normas a través del proceso de revisión de código. El GSS debe guiar el comportamiento del desarrollador a través de la aplicación de normas de codificación con herramientas automatizadas y designando mentores de las mismas. El GSS debe combinar técnicas de evaluación automatizadas con reglas a medida para encontrar problemas de manera eficiente.

[CR2.2] Hacer cumplir las normas de codificación. Una violación de las normas de codificación segura de la organización es motivo suficiente para rechazar un fragmento de código. La revisión de código es objetiva: no se convierte en un debate acerca de si un código erróneo es una vulnerabilidad explotable o no. La parte obligatoria de la norma puede comenzar siendo tan simple como armar una lista de funciones prohibidas. En algunos casos, las normas de codificación se publican como directrices de desarrollo específicas para pilas de tecnología (por ejemplo, directrices para C++ o Spring) y luego se hacen cumplir durante el proceso de revisión de código o directamente en el IDE. Tenga en cuenta que las directrices pueden ser tanto positivas ("hacer esto de esta manera") como negativas ("no utilizar esta API").

[CR2.5] Designar mentores de herramientas. Los mentores están disponibles para mostrar a los desarrolladores cómo obtener el máximo provecho de las herramientas de revisión de código. Si el GSS es habilidoso con las herramientas, puede utilizar el horario laboral para ayudar a los desarrolladores a establecer la configuración correcta o a comenzar con la interpretación de los resultados. Alternativamente, algún integrante del GSS puede trabajar con un equipo de desarrollo durante la primera revisión que este equipo realice. Con el tiempo, se puede distribuir en la organización del desarrollo la utilización de una herramienta centralizada, gestionada por los mentores de herramientas.

[CR2.6] Utilizar herramientas automatizadas con reglas a medida. Personalizar el análisis estático para mejorar la eficiencia y reducir los falsos positivos. Utilizar reglas personalizadas para encontrar errores específicos de las normas de codificación o del *middleware* de la organización. Desactivar los controles que no son relevantes. El mismo grupo mentor de la herramienta probablemente guiará la personalización. Las reglas a medida pueden estar explícitamente vinculadas a la utilización adecuada de las pilas de la tecnología en un sentido positivo, y también evitar los errores habituales encontrados en el código de la empresa en un sentido negativo.

tres

CR Nivel 3: Construir una fábrica de revisión automatizada de código con reglas a medida. El GSS debe construir una capacidad para encontrar y erradicar los errores de programación específicos de la base de código completa.

[CR3.2] Construir una fábrica. Se combinan los resultados de evaluación para que las múltiples técnicas de análisis alimenten un proceso único de elaboración de informes y corrección. El GSS puede escribir *scripts* para invocar automáticamente múltiples técnicas de detección, y combinar los resultados en un formato que pueda ser consumido por una única solución de revisión y generación de reportes. Los motores de análisis pueden combinar análisis estático y dinámico. La parte difícil de esta actividad es normalizar la información de vulnerabilidades proveniente de fuentes dispares que utilizan terminología conflictiva. En algunos casos, un enfoque del tipo CWE puede ayudar con la nomenclatura. Lo cierto es que la combinación de múltiples fuentes ayuda a guiar decisiones de mitigación de riesgo mejor fundamentadas.

[CR3.3] Construir una capacidad para erradicar errores de programación específicos de la base de código. Cuando se encuentra un nuevo tipo de error de programación, el GSS escribe reglas para encontrarlo y las utiliza para identificar todas sus ocurrencias dentro de la base de código. Es posible, así, erradicar totalmente el tipo de error de programación sin esperar a que cada proyecto llegue a la etapa de revisión de código dentro de su ciclo de vida. Una empresa con solo un puñado de aplicaciones de software tendrá mayor facilidad para llevar adelante esta actividad que las empresas con un gran número de aplicaciones de gran tamaño.

[CR3.4] Automatizar la detección de código malicioso. La revisión automatizada de código se utiliza para identificar código peligroso escrito por desarrolladores maliciosos, sean estos empleados internos o proveedores subcontratados. Ejemplos de código malicioso que puede ser blanco de esta revisión incluyen: puertas traseras, bombas lógicas, bombas de tiempo, canales inicuos de comunicación, lógica del programa alterada e inyección de código dinámico. Si bien la mera automatización sin reglas a medida puede identificar algunas construcciones maliciosas genéricas, rápidamente se vuelve una necesidad contar con reglas personalizadas que encaucen el accionar de las herramientas de análisis estático utilizadas para producir patrones de código aceptables e inaceptables en la base de código de la organización. La revisión manual de código utilizada para detectar código malicioso es un buen comienzo, pero es insuficiente para completar esta actividad.

PUNTOS DE CONTACTO CON EL SSDL: Pruebas de Seguridad (ST)

La meta general de la práctica Pruebas de Seguridad es el control de calidad que se llevó a cabo durante el ciclo de desarrollo. Quienes llevan a cabo las pruebas de seguridad deben garantizar la detección y corrección de los errores de programación de seguridad. El GSS debe hacer cumplir la adhesión a las normas y la reutilización de las características de seguridad aprobadas.

uno

ST Nivel 1: Mejorar el aseguramiento de la calidad (QA) más allá de la perspectiva funcional. El área de Aseguramiento de la Calidad (QA) debe avanzar para incluir pruebas funcionales de condiciones de borde y de contorno en sus grupos de casos de prueba. Los grupos de casos de prueba deben incluir pruebas funcionales de seguridad.

[ST1.1] Garantizar que QA soporta pruebas de condiciones de borde y de contorno. El equipo de QA va más allá de las pruebas funcionales para llevar a cabo pruebas básicas de adversario. Sus integrantes investigan casos de borde y condiciones de contorno simples. No se requieren competencias de atacante. Cuando QA comprende el valor de ir más allá de las pruebas funcionales estándares que utilizan entradas aceptables, comienza lentamente a "pensar como un chico malo". Una discusión de las pruebas de condiciones de contorno conduce naturalmente a la noción de un atacante sondeando a propósito los bordes. ¿Qué sucede cuando usted introduce la contraseña incorrecta una y otra vez?

[ST1.3] Conducir pruebas con los requisitos y características de seguridad. Los testers apuntan a los mecanismos de seguridad declarativa derivados de los requisitos y características de seguridad. Por ejemplo, un *tester* podría tratar de acceder a una funcionalidad administrativa como un usuario sin privilegios o verificar que una cuenta de usuario se bloquea después de un cierto número de intentos de autenticación fallidos. En su mayor parte, las características de seguridad se pueden probar de manera similar a otras características del software. También se prueban los mecanismos de seguridad basados en requisitos, tales como el bloqueo de cuenta, las limitaciones en las transacciones, los permisos, y así sucesivamente. Por supuesto que la seguridad del software no es el software de seguridad, pero es fácil comenzar con las características.

dos

ST Nivel 2: Integrar la perspectiva atacante en los planes de prueba. QA debe integrar en su proceso herramientas de pruebas de seguridad de caja negra. Debe construir *suites* de casos de prueba para las características funcionales de seguridad. El GSS debe compartir su conocimiento sobre seguridad y los resultados de las pruebas con QA.

[ST2.1] Integrar herramientas de seguridad de caja negra en el proceso de QA. La organización utiliza una o más herramientas de pruebas de seguridad de caja negra como parte del proceso de aseguramiento de la calidad. Las herramientas son valiosas porque encapsulan la perspectiva del atacante, aunque de una manera genérica. Herramientas como Security AppScan de IBM o WebInspect de HP son relevantes para las aplicaciones web, y marcos de referencia para *fuzzing* como Codenomicon son aplicables a la mayoría de los protocolos de red. En algunas situaciones, otros grupos pueden colaborar con el GSS en la aplicación de las herramientas. Por ejemplo, un equipo de pruebas podría ejecutar la herramienta pero recurrir al GSS para que le ayude a interpretar los resultados. Independientemente de quién ejecuta la herramienta de caja negra, las pruebas deberían estar debidamente integradas en el ciclo de QA del SSDL.

[ST2.4] Compartir los resultados de seguridad con QA. El GSS comparte de forma rutinaria los resultados obtenidos en las revisiones de seguridad con el departamento de QA. Con el tiempo, los ingenieros de QA asimilan fundamentos de la mentalidad propia a la seguridad. Utilizar los resultados de seguridad para informar y hacer evolucionar patrones particulares de prueba puede ser un poderoso mecanismo que conduce a la realización de mejores pruebas de seguridad. Esta actividad se beneficia de un abordaje ingenieril, fuertemente técnico, de la función de QA.

tres

ST Nivel 3: Entregar pruebas de seguridad basadas en riesgos. QA debe incluir pruebas de seguridad en las suites de pruebas de regresión automatizadas. El GSS debe garantizar que la realización de este tipo de pruebas de seguridad y su profundidad está guiada por el conocimiento de la base de código y de sus riesgos asociados, como así también la realización de pruebas de adversario que simulan la perspectiva del atacante.

[ST3.1] Incluir pruebas de seguridad en la automatización de QA. Las pruebas de seguridad se realizan junto con las pruebas funcionales como parte de las pruebas de regresión automatizadas. El mismo marco de referencia de automatización alberga a ambas. Las pruebas de seguridad forman parte de la rutina. Se pueden conducir a partir de los casos de abuso identificados previamente, durante el ciclo de vida, o pueden derivar de ajustar creativamente las pruebas funcionales.

[ST3.2] Realizar pruebas *fuzz* a medida para las API de aplicación. Los ingenieros de automatización de pruebas personalizan un marco de referencia de *fuzzing* para las API de la organización. Pueden comenzar desde cero o utilizar una herramienta de *fuzzing* existente, pero en los dos casos la personalización va más allá de crear descripciones de protocolo o plantillas de formato de archivo personalizadas. El marco de referencia de *fuzzing* tiene un conocimiento embebido de las interfaces de aplicación que se invocan. Las colecciones de software y datos utilizados en las pruebas desarrolladas explícitamente para aplicaciones particulares pueden ser buenos lugares donde integrar las pruebas fuzz.

[ST3.3] Conducir las pruebas con resultados del análisis de riesgo. Los testers utilizan los resultados del análisis de la arquitectura para dirigir su trabajo. Por ejemplo, si el análisis de la arquitectura concluye que "la seguridad del sistema depende de que las transacciones sean atómicas y no se interrumpan sin terminar", entonces las transacciones se convertirán en el blanco principal de las pruebas de adversario. Se pueden desarrollar pruebas de adversario como éstas de acuerdo al perfil de riesgo; ubicar, en primer lugar, los defectos de alto riesgo.

[ST3.4] Aprovechar el análisis de cobertura. Los *testers* miden la cobertura de código de sus pruebas de seguridad con el fin de identificar el código que no está siendo chequeado. La cobertura de código conduce a una mayor profundidad de las pruebas de seguridad. Las herramientas de prueba de caja negra estándares logran una cobertura excepcionalmente baja, dejando la mayor parte del software bajo prueba sin explorar. No deje que esto suceda con sus pruebas. Es bueno hacer uso de mediciones estándares para la cobertura, tales como cobertura de funciones, cobertura de líneas de código o cobertura de condiciones múltiples.

[ST3.5] Comenzar a construir y aplicar pruebas de seguridad de adversario (casos de abuso). Las pruebas comienzan a incorporar casos de prueba basados en casos de abuso que proporciona el GSS. Los testers van más allá de verificar la funcionalidad y asumen la perspectiva del atacante. Por ejemplo, los testers pueden tratar de replicar sistemáticamente los incidentes partiendo de la historia de la organización. Los casos de abuso y de mal uso basados en la perspectiva del atacante también pueden estar guiados por políticas de seguridad, inteligencia de ataque y directrices. Esto implica ir más allá de las pruebas de características para intentar romper el software que se está probando.

DESPLIEGUE: Pruebas de Penetración (PT)

La meta general de la práctica Pruebas de Penetración es el control de calidad del software implementado. Quienes llevan a cabo las pruebas de penetración deben garantizar que se logre la detección y corrección de los defectos de seguridad. El GSS debe hacer cumplir la adhesión a las normas y exigir que se reutilicen las características de seguridad aprobadas.

uno

PT Nivel 1: Remediar los resultados de las pruebas de penetración. Los gerentes y el GSS deben iniciar el proceso de pruebas de penetración, ya sea con recursos internos o externos. Deben garantizar que las deficiencias descubiertas son tratadas y que todo el mundo tenga conciencia del progreso en esta materia.

[PT1.1] Utilizar *testers* de penetración externos para encontrar problemas. Muchas organizaciones no están dispuestas a tratar la seguridad del software hasta que no haya evidencia inequívoca de que la organización no es mágicamente inmune a los problemas en este campo. Si la seguridad no ha sido una prioridad, los *testers* de penetración externos pueden demostrar que el código de la organización necesita una revisión. Se puede contratar *testers* de penetración para romper una aplicación de alto perfil y así dejarla en evidencia. Con el tiempo, el foco de las pruebas de penetración pasa de un "te dije que nuestra aplicación estaba rota" a una prueba de humo y un chequeo de sanidad que son realizados antes de su lanzamiento. Los testers de penetración externos aportan una mirada nueva sobre el problema.

[PT1.2] Alimentar el sistema de gestión de defectos y mitigación con los resultados. Los resultados de las pruebas de penetración retroalimentan al proceso de desarrollo mediante la gestión de defectos o los canales de mitigación establecidos, y el desarrollo responde utilizando su gestión de defectos y su proceso de lanzamiento. El ejercicio demuestra la capacidad de la organización para mejorar el estado de la seguridad. Muchas empresas están empezando a destacar la importancia crítica de, no solo identificar sino, más importante, solucionar los problemas de seguridad. Una forma de garantizar la continuidad de la atención es añadir una indicación como, por ejemplo, "bandera de seguridad" al sistema de seguimiento de errores de programación y gestión de defectos. La evolución de desarrolladores y operadores, así como la de las estructuras integradas de equipo, no elimina la necesidad de contar con sistemas formales de gestión de defectos.

[PT1.3] Utilizar internamente herramientas de pruebas de penetración. La organización crea una capacidad interna que le permite efectuar pruebas de penetración que hacen uso de herramientas. Esta capacidad puede pertenecer al GSS o a un equipo especializado y capacitado que provenga de otra parte de la organización. Las herramientas mejoran la eficiencia y la repetitividad del proceso de pruebas; pueden incluir productos enlatados, herramientas de penetración de red estándares que comprenden a la capa de aplicaciones y *scripts* escritos a mano.

dos

PT Nivel 2: Programar pruebas de penetración en forma regular realizadas por testers de penetración internos instruidos. El GSS debe crear la capacidad de efectuar pruebas de penetración internas que se apliquen periódicamente a todas las aplicaciones, y debe compartir sus conocimientos de seguridad y los resultados de las pruebas con todos los *testers* de penetración.

[PT2.2] Proporcionar a los *testers* de penetración toda la información disponible. Los *testers* de penetración, ya sean internos o externos, utilizan toda la información disponible acerca de su blanco. Los *testers* de penetración pueden hacer un análisis más profundo y encontrar los problemas más interesantes cuando tienen el código fuente, los documentos de diseño, los resultados del análisis de la arquitectura y los resultados de revisión de código. Se debe dar a los *testers* de penetración todo el material que se creó a lo largo del SSDL. Si su *tester* de penetración no le solicita el código, es necesario reemplazarlo.

[PT2.3] Programar pruebas de penetración periódicas para cobertura de la aplicación. Se prueban periódicamente todas las aplicaciones dentro del alcance del GSS según un cronograma establecido (que puede estar vinculado al calendario o al ciclo de lanzamiento). Las pruebas sirven como un chequeo de salud y ayudan a garantizar que el software de ayer no es vulnerable a los ataques de hoy. Las aplicaciones de alto perfil deberían atravesar una prueba de penetración por lo menos una vez al año. Un aspecto importante de las pruebas periódicas es garantizar que los problemas identificados en una prueba de penetración realmente se corrijan y evitar, de modo certero, que sean introducidos de nuevo.

tres

PT Nivel 3: Llevar a cabo pruebas de penetración profundas. Los gerentes deben garantizar que el conocimiento de las pruebas de penetración de la organización mantiene el ritmo de perfeccionamiento de los atacantes. El GSS debe aprovechar el conocimiento de la organización para personalizar las herramientas de pruebas de penetración.

[PT3.1] Utilizar testers de penetración externos para llevar a cabo un análisis en profundidad. La organización utiliza testers de penetración externos para hacer un análisis en profundidad, con el fin de controlar proyectos importantes y para introducir nuevas ideas en el GSS. Estos testers son expertos y especialistas. Mantienen la organización actualizada con la última versión de la perspectiva del atacante y tienen un registro de casos exitosos en quebrantar la seguridad del tipo de software que se está probando. Los testers de penetración expertos siempre quebrarán un sistema. La pregunta es si con ello generarán nuevas formas de pensar acerca de los ataques, las que pueden ser útiles para diseñar, implementar y fortalecer los sistemas nuevos. La creación de tipos nuevos de ataques a partir de la inteligencia de amenazas y casos de abuso evita enfoques conducidos por listas de comprobación que sólo buscan tipos de problemas conocidos.

[PT3.2] ¿El GSS personalizó herramientas y scripts para las pruebas de penetración? El GSS crea herramientas de pruebas de penetración o adapta herramientas públicas con las que se pueda atacar de manera más eficiente e integral los sistemas de la organización. Las herramientas mejoran la eficiencia del proceso de pruebas de penetración, sin sacrificar por ello la profundidad de los problemas que puede identificar el GSS. Las herramientas que se pueden adaptar siempre son preferibles a las genéricas.

DESPLIEGUE: Entorno del Software (SE)

La meta general de la práctica Entorno del Software es la gestión de cambios. Quienes son responsables del entorno del software deben garantizar su capacidad para realizar cambios autorizados y detectar los cambios y actividad no autorizados. Los gerentes deben hacer cumplir la adhesión a la política corporativa.

uno

SE Nivel 1: Garantizar que el entorno de aplicación soporta la seguridad del software. El grupo de operaciones garantiza que están funcionando los controles de seguridad de red y de equipos de cómputo, y monitoriza de manera proactiva el software, incluidas las entradas de datos a la aplicación.

[SE1.1] Monitorizar la entrada de datos de las aplicaciones. La organización monitoriza la entrada de datos al software que ejecuta, con el fin de detectar ataques. Para el código web, el trabajo lo puede hacer un *firewall* de aplicación web (WAF). El GSS puede ser responsable del cuidado y alimentación del sistema; responder al ataque no formaría parte de esta actividad. Los WAF pasivos que escriben registros de actividad (*logs*) pueden ser útiles si alguien revisa periódicamente dichos registros. Por otro lado, si el WAF no es monitorizado, cuando una aplicación caiga en el bosque no hará ruido.

[SE1.2] Garantizar que se encuentra implementada la seguridad básica en los equipos de computación y redes. La organización proporciona una base sólida al software al garantizar que se encuentra implementada la seguridad en los equipos de computación y en las redes. Es común que los equipos de seguridad de operaciones sean responsables de tareas como la aplicación de parches de sistemas operativos y el mantenimiento de firewalls. Construir la seguridad del software antes que la de la red es como ponerse los pantalones antes que la ropa interior.

dos

SE Nivel 2: Utilizar guías de instalación publicadas y firma de código. El GSS debe garantizar que los procesos de desarrollo de software protegen la integridad del código. El GSS debe, además, garantizar la creación de guías de instalación y mantenimiento de aplicaciones que puedan ser utilizadas por el grupo de operaciones.

[SE2.2] Publicar guías de instalación. El SSDL requiere crear una guía de instalación que ayude a los operadores a instalar y configurar el software de manera segura. Si se requiere de pasos especiales para garantizar que una implementación es segura, estos pasos se describen en la guía de instalación. También debería incluir una discusión de los componentes de software comercial enlatado. En algunos casos, las guías de instalación se distribuyen a los clientes que compran el software. La evolución de la interacción entre los equipos de desarrollo y de TI, así como la de las estructuras de equipo integrados, no elimina la necesidad de escribir las guías. Por supuesto, la seguridad por defecto es siempre el mejor camino a seguir.

[SE2.4] Utilizar firma de código. La organización utiliza la firma de código para publicar el software que cruza fronteras de confianza. La firma de código es muy útil para proteger la integridad del software que queda fuera del control de la organización, como las aplicaciones empaquetadas o clientes gruesos. El hecho de que algunas plataformas móviles requieran que el código de aplicación esté firmado no indica que se haga un uso institucional de la firma de código.

tres

SE Nivel 3: Proteger el código del cliente y monitorizar el comportamiento del software. El GSS debe garantizar que todo el código que abandona la organización está protegido. El grupo de operaciones debe monitorizar el comportamiento del software.

[SE3.2] Utilizar protección de código. Con el fin de proteger la propiedad intelectual y hacer más difícil la explotación de una vulnerabilidad, la organización crea barreras para la ingeniería inversa. Se podrían aplicar técnicas de ofuscación como parte del proceso de compilación y lanzamiento. El empleo de controles específicos de la plataforma como *Data Execution Prevention* (DEP), *Safe Structured Error Handling* (SafeSEH) y *Address Space Layout Randomization* (ASLR) puede hacer más difícil el desarrollo de un mecanismo de explotación de vulnerabilidad.

[SE3.3] Monitorizar y diagnosticar el comportamiento de la aplicación. La organización monitoriza el comportamiento del software en producción en busca de malos comportamientos y signos de ataque. Esta actividad va más allá de monitorizar el equipo de computación y la red, y busca determinar la existencia de problemas que son específicos del software, como indicios de fraude. Los sistemas de detección de intrusiones y de detección de anomalías a nivel de la aplicación se pueden centrar en la interacción de una aplicación con el sistema operativo (a través de llamadas al sistema) o en los tipos de datos que una aplicación consume, origina y manipula.

DESPLIEGUE: Gestión de Configuración y Gestión de Vulnerabilidades (CMVM)

La meta general de la práctica Gestión de Configuración y Gestión de Vulnerabilidades es la gestión de cambios. El GSS y los responsables de cada aplicación deben garantizar su capacidad de llevar a cabo el seguimiento de los cambios autorizados y detectar los cambios y actividad no autorizados. Los responsables de aplicaciones deben hacer cumplir la adhesión a la política corporativa.

uno

CMVM Nivel 1: Utilizar datos de monitorización de las operaciones para conducir el comportamiento del desarrollador. El GSS debe apoyar la respuesta a incidentes. El GSS debe utilizar datos de las operaciones para sugerir cambios en el SSDL y en el comportamiento del desarrollador.

[CMVM1.1] Crear o interactuar con la respuesta a incidentes. El GSS está preparado para responder a un incidente. El grupo crea su propia capacidad de respuesta a incidentes o interactúa con el equipo de respuesta a incidentes existente en la organización. Una reunión regular entre el GSS y el equipo de respuesta a incidentes puede mantener un flujo de información en ambas direcciones. En muchos casos, las iniciativas de seguridad del software evolucionaron a partir de estos equipos de respuesta a incidentes, que fueron quienes comenzaron a comprender que las vulnerabilidades de software eran la ruina de su existencia.

[CMVM1.2] Identificar los defectos de software encontrados en la monitorización de las operaciones y retroalimentar al desarrollo. Los defectos identificados a través de la monitorización de las operaciones retroalimentan al proceso de desarrollo y se utilizan para cambiar el comportamiento del desarrollador. Los contenidos de los registros de actividades de producción pueden ser reveladores (o pueden revelar la necesidad de mejorarlos), En algunos casos, parece dar resultados disponer de una manera de ingresar datos, evaluados y ordenados por prioridad, de incidentes en un sistema de seguimiento de errores de programación (muchas veces utilizando una etiqueta específica para los errores de seguridad). La idea es cerrar el lazo de flujo de información y estar seguros de que se corrijan los problemas de seguridad. En el mejor de los casos, los procesos en el SSDL pueden ser mejorados en base a los datos operacionales.

dos

CMVM Nivel 2: Garantizar que la respuesta a la emergencia esté disponible durante ataques a la aplicación. Cuando hay en curso un ataque contra alguna aplicación, los gerentes y el GSS deben apoyar al grupo encargado de dar respuesta a la emergencia. Los gerentes y el GSS deben mantener un inventario de código. El GSS utiliza datos de operaciones para dirigir la evolución del SSDL y el comportamiento del desarrollador.

[CMVM2.1] Tener respuesta, ante una emergencia, para la base de código. La organización puede hacer cambios rápidos de código cuando una aplicación está bajo ataque. Un equipo de respuesta rápida trabaja en conjunto con los responsables de las aplicaciones y el GSS para estudiar el código y el ataque, encontrar una solución y aplicar un parche en el ambiente de producción. Muchas veces, el equipo de respuesta a la emergencia es el propio equipo de desarrollo. Simulacros de incendio no cuentan: aquí se requiere de un proceso bien definido.

[CMVM2.2] Realizar el seguimiento y la corrección de los errores de programación de software que se encuentran en las operaciones. Los defectos que se encuentran en operaciones retroalimentan al proceso de desarrollo; son ingresados en sistemas establecidos de gestión de defectos y se realiza su seguimiento a lo largo de la fase de corrección. Este mecanismo puede tomar la forma de un puente de doble vía que comunique a buscadores de errores de programación con quienes los corrigen. Se debe garantizar que el ciclo se cierra completamente. Por otra parte, establecer una etiqueta especial para seguridad en el sistema de seguimiento de errores de programación puede facilitar el rastreo.

[CMVM2.3] Desarrollar un inventario operativo de las aplicaciones. La organización cuenta con un mapa de sus implementaciones de software. Si se debe cambiar una porción de código, el grupo encargado de operaciones puede identificar de manera fiable todos los lugares donde se debe implantar el cambio. A veces se destacan los componentes comunes compartidos entre varios proyectos, de manera que cuando se produzca un error en una aplicación, también se pueda corregir otras aplicaciones que comparten los mismos componentes.

tres

CMVM Nivel 3: Crear un ciclo estrecho entre operaciones y desarrollo. El GSS debe garantizar que el SSDL aborda las deficiencias de código que se encontraron en las operaciones e incluye mejoras que eliminan las causas de fondo.

[CMVM3.1] Corregir todas las ocurrencias de errores de programación de software que se encuentran durante las operaciones. La organización corrige todas las instancias de cada error de programación de software que se encuentran durante las operaciones, no solo el pequeño número de instancias que activaron los informes de errores. Esto requiere de la capacidad de reexaminar toda la base de código cuando salen a la luz nuevos tipos de errores de programación (véase [CR3.3 Construir una capacidad para erradicar errores de programación específicos de la base de código]). Una manera de abordar este punto es crear un conjunto de reglas que generalizan un error de programación desplegado en algo que se puede examinar cuando se realiza una revisión automatizada de código.

[CMVM3.2] Mejorar el SSDL para evitar errores de programación de software que se encuentran durante las operaciones. La experiencia de las operaciones conduce a cambios en el SSDL. El SSDL se refuerza para evitar la reintroducción de los errores de programación que se encontraron durante las operaciones. Para hacer este proceso más sistemático, la respuesta a incidentes post mórtem puede incluir un paso que sea "retroalimentar al SSDL". Esto funciona mejor cuando el análisis de la causa de fondo indica en qué lugar del SDLC se pudo introducir un error o dónde se deslizó sin ser detectado. Un enfoque ad hoc no es suficiente.

[CMVM3.3] Simular crisis del software. El GSS simula crisis de seguridad del software de alto impacto para garantizar capacidades de respuesta a incidentes de software que minimicen los daños. Las simulaciones pueden poner a prueba la capacidad de identificar y mitigar amenazas específicas o, en otros casos, partir de la suposición de que un sistema o servicio crítico ya está comprometido y evaluar la capacidad de la organización para responder a esa situación. Cuando las simulaciones modelan ataques exitosos, una cuestión importante a considerar es el período de tiempo requerido para poner las cosas en orden. En cualquier caso, las simulaciones se deben centrar en fallas del software relevantes a la seguridad y no en desastres naturales u otros tipos de ejercicios de respuesta a emergencias. Si el centro de datos está ardiendo, el GSS no será uno de los primeros equipos en responder.

[CMVM3.4] Ejecutar un programa de recompensas para el descubrimiento de errores que impactan en la seguridad del software. La organización solicita a investigadores externos informes de vulnerabilidades y paga una recompensa por cada vulnerabilidad informada que se verifica y acepta. Las recompensas generalmente siguen un gradiente asociado a múltiples factores, tales como tipo de vulnerabilidad (por ejemplo, ejecución remota de código se valúa en U$S 10.000 mientras que *cross-site request forgery* -CSRF- se valúa en U$S 750), factibilidad de su explotación (exploits basados en comandos demostrables se valúan aún más), o servicios específicos o versiones de software (servicios ampliamente desplegados o críticos garantizan pagos mayores). Actividades ad hoc o de corta duración, tales como desafíos o competencias, no se tienen en cuenta. [Esta es una actividad nueva que se informará en BSIMM6.]

El Esqueleto BSIMM

El Esqueleto BSIMM proporciona una vista rápida del modelo de madurez y es útil a la hora de evaluar un programa de seguridad del software. El esqueleto incluye una página por cada práctica, organizada cada una de ellas en tres niveles. Cada actividad está asociada a un objetivo. Descripciones más completas de las actividades, ejemplos y la definición de los términos se pueden encontrar en el documento principal.

GOBERNANZA: ESTRATEGIA Y MÉTRICAS Planificación, asignación de roles y responsabilidades, identificación de metas de seguridad del software, determinación de presupuestos, identificación de métricas y puertas de control.		
Objetivo	**Actividad**	**Nivel**
[SM1.1] explicitar el plan	publicar el proceso (roles, responsabilidades, plan), evolucionar a medida que sea necesario	1
[SM1.2] construir el soporte en toda la organización	crear el rol de promotor y realizar el marketing interno	
[SM1.3] construir el soporte en toda la organización	educar a los ejecutivos	
[SM1.4] establecer las puertas de control del SSDL (pero no hacerlas obligatorias)	identificar la ubicación para las puertas de control (gates), reunir los artefactos necesarios	
[SM1.6] poner en claro quién toma el riesgo	requerir la aprobación de la seguridad	
[SM2.1] fomentar la transparencia (o la competencia)	publicar internamente datos sobre la seguridad del software	2
[SM2.2] cambiar el comportamiento	cumplir obligatoriamente las puertas de control satisfaciendo mediciones y realizar el seguimiento de las excepciones	
[SM2.3] crear una base amplia de soporte	crear o hacer crecer un satélite	
[SM2.5] definir el éxito	identificar métricas y utilizarlas para guiar los presupuestos	
[SM3.1] saber dónde están todas las aplicaciones en el inventario	utilizar una aplicación de seguimiento interno con una visión de cartera	3
[SM3.2] crear un soporte externo	ejecutar un programa de marketing externo	

GOBERNANZA: CUMPLIMIENTO Y POLÍTICA

Identificar los controles para los regímenes de cumplimiento, desarrollar los controles contractuales (SLA de software comercial enlatado), establecer la política de la organización, auditar contra la política.

	Objetivo	Actividad	Nivel
[CP1.1]	comprender los requisitos de cumplimiento (FFIEC, GLBA, OCC, PCI, SOX, HIPAA)	unificar las presiones regulatorias	1
[CP1.2]	promover la privacidad	identificar obligaciones relativas a la información de identificación personal (IIP)	
[CP1.3]	satisfacer las necesidades regulatorias o la demanda de los clientes con un enfoque unificado	crear una política	
[CP2.1]	promover la privacidad	identificar el inventario de datos de IIP	2
[CP2.2]	garantizar la responsabilidad por el riesgo del software	requerir la aprobación de seguridad para los riesgos relacionados con el cumplimiento	
[CP2.3]	alinear las prácticas con el cumplimiento	implementar y realizar el seguimiento de los controles para el cumplimiento	
[CP2.4]	garantizar que los proveedores no comentan errores de cumplimiento	incluir en todos los contratos con proveedores acuerdos de nivel de servicio de seguridad del software (SLA)	
[CP2.5]	interesar a los ejecutivos en los beneficios	promover la conciencia ejecutiva acerca de las obligaciones de cumplimiento y privacidad	
[CP3.1]	demostrar el historial de cumplimiento	crear documentación atractiva para reguladores	3
[CP3.2]	gestionar a proveedores	imponer políticas a proveedores	
[CP3.3]	mantener las políticas alineadas con la realidad	conducir la retroalimentación de datos desde el SSDL hacia las políticas	

	GOBERNANZA: CAPACITACIÓN		
	Objetivo	**Actividad**	**Nivel**
[T1.1]	promover la cultura de la seguridad en toda la organización	proveer capacitación orientada a la concientización	1
[T1.5]	desarrollar capacidades más allá de la concientización	proporcionar contenidos avanzados específicos para el rol (herramientas, pilas de tecnología, ranking de errores de programación)	
[T1.6]	verse a sí mismo en el problema	crear y utilizar material específico sobre la historia de la empresa	
[T1.7]	reducir el impacto sobre los destinatarios de la capacitación y desarrollar un equipo de capacitación	proporcionar capacitación individual a demanda	
[T2.5]	educar/fortalecer una red social	mejorar el satélite a través de capacitación y eventos	2
[T2.6]	garantizar que las incorporaciones de personal mejoren la cultura	incluir recursos de seguridad en la etapa de inducción	
[T2.7]	crear una red social vinculada con el desarrollo	identificar el satélite a través de la capacitación	
[T3.1]	alinear la cultura de la seguridad con la carrera	premiar el progreso logrado en la capacitación (certificación o RRHH)	3
[T3.2]	extender la cultura de la seguridad a los proveedores	capacitar a los proveedores o empleados subcontratados	
[T3.3]	promocionar la cultura de la seguridad como diferenciador	organizar eventos externos relacionados con la seguridad del software	
[T3.4]	mantener actualizado al equipo y abordar la rotación	exigir un curso de actualización anual	
[T3.5]	actuar como recurso informal para aprovechar momentos de enseñanza	establecer horarios de oficina del GSS	

INTELIGENCIA: MODELOS DE ATAQUE

Modelado de amenaza, casos de abuso, clasificación de datos, patrones de ataques para tecnologías específicas.

	Objetivo	Actividad	Nivel
[AM1.1]	comprender conceptos básicos sobre ataques	construir y mantener una lista de los N principales, posibles ataques	1
[AM1.2]	priorizar las aplicaciones por los datos consumidos/manipulados	crear un esquema de clasificación e inventario de datos	
[AM1.3]	comprender el "quién" de los ataques	identificar a potenciales atacantes	
[AM1.4]	comprender la historia de la organización	recopilar y publicar historias de ataques	
[AM1.5]	estar actualizado sobre el entorno de ataques/vulnerabilidades	reunir inteligencia sobre ataques	
[AM1.6]	comunicar la perspectiva del atacante	crear un foro interno para discutir los ataques	
[AM2.1]	proporcionar recursos para pruebas de seguridad y de AA	construir patrones de ataque y casos de abuso ligados a posible atacantes	2
[AM2.2]	comprender ataques orientados a la tecnología	crear patrones de ataque para tecnologías específicas	
[AM3.1]	llevar la delantera de la curva de ataque	tener un equipo científico que desarrolle nuevos métodos de ataque	3
[AM3.2]	proporcionar armas a los testers y auditores	crear y emplear métodos automatizados para hacer lo que harán los atacantes	

INTELIGENCIA: CARACTERÍSTICAS DE SEGURIDAD Y DISEÑO

Patrones de seguridad para los principales controles de seguridad, marcos de referencia "middleware" para los controles, orientación proactiva de seguridad.

	Objetivo	Actividad	Nivel
[SFD1.1]	crear orientación proactiva en base a las características de seguridad	construir y publicar las características de seguridad	1
[SFD1.2]	incorporar el pensamiento de seguridad en el grupo de arquitectura	involucrar al GSS con la arquitectura	
[SFD2.1]	crear un diseño proactivo de seguridad en base a pilas de tecnología	construir marcos de referencia "middleware" y bibliotecas comunes, seguros por diseño (T: revisión de código)	2
[SFD2.2]	atender la necesidad para la nueva arquitectura	crear la capacidad en el GSS de resolver problemas difíciles de diseño	
[SFD3.1]	formalizar el consenso respecto al diseño	formar un consejo o comité central de revisión para aprobar y mantener los patrones de diseño seguros	3
[SFD3.2]	promover la eficiencia del diseño	exigir la utilización de características de seguridad y marcos de referencia aprobados (T:AA)	
[SFD3.3]	reutilizar la práctica	encontrar y publicar patrones ya maduros de diseño de la organización	

INTELIGENCIA: NORMAS Y REQUISITOS

Explicitar los requisitos de seguridad, software comercial enlatados recomendados, normas para los principales controles de seguridad, normas para las tecnologías en uso, comité de revisión de normas.

	Objetivo	Actividad	Nivel
[SR1.1]	satisfacer la demanda para las características de seguridad	crear normas de seguridad (T: características/diseño de seguridad)	1
[SR1.2]	garantizar que todos conocen dónde obtener lo último y mejor	crear un portal de seguridad	
[SR1.3]	estrategia de cumplimiento	traducir las restricciones de cumplimiento en requisitos	
[SR1.4]	indicar a las personas qué buscar en la revisión de código	utilizar normas de codificación segura	
[SR2.2]	formalizar el proceso de elaboración de normas	crear un comité de revisión de normas	2
[SR2.3]	reducir la carga de trabajo del GSS	crear normas para pilas de tecnología	
[SR2.4]	gestionar el riesgo del código abierto	identificar el código abierto	
[SR2.5]	interesar en el beneficio a partir del departamento legal y el enfoque normalizado	crear un modelo de SLA (T: cumplimiento y política)	
[SR3.1]	gestionar el riesgo del código abierto	control del riesgo del código abierto	3
[SR3.2]	educar a los proveedores	comunicar las normas a los proveedores	

PUNTOS DE CONTACTO CON EL SSDL: ANÁLISIS DE LA ARQUITECTURA

Capturar diagramas de arquitectura de software, aplicar listas de riesgos y amenazas, adoptar un proceso de revisión, construir un plan de evaluación y recuperación.

	Objetivo	Actividad	Nivel
[AA1.1]	comenzar con AA	llevar a cabo la revisión de las características de seguridad	1
[AA1.2]	demostrar el valor de AA con datos reales	llevar a cabo la revisión del diseño para aplicaciones de alto riesgo	
[AA1.3]	crear una capacidad interna sobre la arquitectura de seguridad	tener al GSS como líder de los esfuerzos de revisión	
[AA1.4]	tener un enfoque ligero para clasificar y priorizar los riesgos	utilizar un cuestionario sobre riesgos de diseño para clasificar las aplicaciones	
[AA2.1]	modelar objetos	definir y utilizar el proceso de análisis de la arquitectura	2
[AA2.2]	promover un lenguaje común para describir la arquitectura	normalizar las descripciones arquitectónicas (incluido el flujo de datos)	
[AA2.3]	construir la capacidad en toda la organización	disponer del GSS como recurso o mentor del análisis de la arquitectura	
[AA3.1]	construir capacidades en toda la organización	contar con arquitectos de software que lideren los esfuerzos de revisión del diseño	3
[AA3.2]	construir una arquitectura proactiva de seguridad	traducir los resultados del análisis a patrones de arquitectura estándares (T: características/diseño)	

PUNTOS DE CONTACTO CON EL SSDL: REVISIÓN DE CÓDIGO

Utilizar herramientas de revisión de código, desarrollar reglas personalizadas, perfiles para el uso de herramientas por diferentes roles, análisis manual, clasificación/evaluación de resultados

	Objetivo	Actividad	Nivel
[CR1.1]	conocer cuáles errores de programación le interesan	crear una lista de los N principales errores de programación (preferir datos reales) (T: capacitación)	1
[CR1.2]	revisar las aplicaciones de alto riesgo de manera oportuna	hacer que el GSS lleve a cabo revisiones ad hoc	
[CR1.4]	impulsar la eficiencia/consistencia con automatización	utilizar herramientas automatizadas junto con la revisión manual	
[CR1.5]	encontrar errores de programación de manera temprana	hacer obligatoria la revisión de código para todos los proyectos	
[CR1.6]	conocer cuáles errores de programación le interesan (para capacitación)	utilizar informes centralizados para cerrar el ciclo de conocimiento y conducir la capacitación (T: estrategia/ métricas)	
[CR2.2]	orientar el comportamiento de manera objetiva	hacer cumplir las normas de codificación	2
[CR2.5]	hacer un uso más eficiente de las herramientas	designar mentores de herramientas	
[CR2.6]	impulsar la eficiencia/reducción de falsos positivos	utilizar herramientas automatizadas con reglas a medida	
[CR3.2]	combinar las técnicas de evaluación	construir una fábrica	3
[CR3.3]	manejar nuevas clases de errores de programación en la base de código ya revisada	construir una capacidad para erradicar errores de programación específicos de la base de código	
[CR3.4]	abordar amenazas internas desde el desarrollo	automatizar la detección de código malicioso	

PUNTOS DE CONTACTO CON EL SSDL: PRUEBAS DE SEGURIDAD

Utilizar herramientas de caja negra para seguridad en QA, pruebas de caja blanca guiadas por el riesgo, aplicación del modelo de ataque, análisis de cobertura de código.

	Objetivo	Actividad	Nivel
[ST1.1]	ejecutar pruebas de adversario más allá de las funcionales	garantizar que QA soporta pruebas de condiciones de borde y de contorno	1
[ST1.3]	comenzar las pruebas de seguridad en un territorio funcional familiar	conducir pruebas con los requisitos y características de seguridad	
[ST2.1]	utilizar la perspectiva encapsulada del atacante	integrar herramientas de seguridad de caja negra en el proceso de QA	2
[ST2.4]	facilitar la mentalidad de seguridad	compartir los resultados de seguridad con QA	
[ST3.1]	incluir pruebas de seguridad en las pruebas de regresión	incluir pruebas de seguridad en la automatización de QA	3
[ST3.2]	instruir en herramientas acerca de su código	realizar pruebas fuzz a medida para las API de aplicación	
[ST3.3]	investigar las demandas del riesgo de manera directa	conducir las pruebas con resultados del análisis de riesgo	
[ST3.4]	orientar las pruebas en profundidad	aprovechar el análisis de cobertura	
[ST3.5]	ir más allá de las pruebas funcionales hacia la perspectiva del atacante	comenzar a construir y aplicar pruebas de seguridad de adversario (casos de abuso)	

DESPLIEGUE: PRUEBAS DE PENETRACIÓN Vulnerabilidades en la configuración final, retroalimentar a la gestión de defectos y mitigación.		
Objetivo	**Actividad**	**Nivel**
[PT1.1] demostrar que el código de su organización también necesita ayuda	utilizar testers de penetración externos para encontrar problemas	1
[PT1.2] corregir lo que encuentra para demostrar el progreso real	alimentar el sistema de gestión de defectos y mitigación con los resultados (T: gestión de configuración/vulnerabilidades)	
[PT1.3] crear una capacidad interna	utilizar internamente herramientas de pruebas de penetración	
[PT2.2] promover un análisis más profundo	proporcionar a los testers de penetración toda la información disponible	2
[PT2.3] verificar la sanidad de manera constante	programar pruebas de penetración periódicas para cobertura de la aplicación	
[PT3.1] estar actualizado con la perspectiva del atacante	utilizar testers de penetración externos para llevar a cabo un análisis en profundidad	3
[PT3.2] automatizar para la eficiencia sin perder profundidad	¿el GSS personalizó herramientas y scripts para las pruebas de penetración?	

DESPLIEGUE: ENTORNO DEL SOFTWARE

Aplicar parches al sistema operativo y la plataforma, firewalls de aplicaciones web, documentación de instalación y configuración, monitorización de aplicación, gestión de cambios, firma de código.

	Objetivo	Actividad	Nivel
[SE1.1]	observar el software	monitorizar la entrada de datos de las aplicaciones	1
[SE1.2]	proporcionar una base sólida de equipos de computación/redes para el software	garantizar que se encuentra implementada la seguridad básica en los equipos de computación y redes	
[SE2.2]	orientar las operaciones en base a las necesidades de la aplicación	publicar guías de instalación	2
[SE2.4]	proteger las aplicaciones (o partes de ellas) que se publican más allá de las fronteras de confianza	utilizar firma de código	
[SE3.2]	proteger la propiedad intelectual y hacer más difícil la explotación de vulnerabilidades	utilizar protección de código	3
[SE3.3]	observar el software	monitorizar y diagnosticar el comportamiento de la aplicación	

DESPLIEGUE: GESTIÓN DE CONFIGURACIÓN Y GESTIÓN DE VULNERABILIDADES

Aplicar parches y actualizar las aplicaciones, control de versión, seguimiento de defectos y mitigación, manejo de incidentes.

	Objetivo	Actividad	Nivel
[CMVM1.1]	saber qué hacer cuando ocurre algo malo	crear o interactuar con la respuesta a incidentes	1
[CMVM1.2]	utilizar datos de las operaciones para cambiar el comportamiento del desarrollo	identificar los defectos de software encontrados en la monitorización de las operaciones y retroalimentar al desarrollo	
[CMVM2.1]	ser capaces de corregir las aplicaciones cuando están bajo ataque directo	tener respuesta a emergencias para la base de código	2
[CMVM2.2]	utilizar datos de las operaciones para cambiar el comportamiento del desarrollo	realizar el seguimiento a la corrección de los errores de programación de software que se encuentran en las operaciones	
[CMVM2.3]	saber dónde está el código	desarrollar un inventario operativo de las aplicaciones	
[CMVM3.1]	aprender de la experiencia operacional	corregir todas las ocurrencias de errores de programación de software que se encuentran durante las operaciones (T: revisión de código)	3
[CMVM3.2]	utilizar datos de las operaciones para cambiar el comportamiento del desarrollo	mejorar el SSDL para evitar errores de programación de software que se encuentran durante las operaciones	
[CMVM3.3]	garantizar que se aplican los procesos para minimizar el impacto de incidentes de software	simular crisis del software	
[CMVM3.4]	solicitar a investigadores externos el descubrimiento de vulnerabilidades	ejecutar un programa de recompensas para el descubrimiento de errores que impactan en la seguridad del software	

Ranking de actividades BSIMM-V

La elección de cuáles de las actividades BSIMM adoptar y en qué orden hacerlo puede resultar un desafío. Sugerimos, en esta situación, crear una estrategia y un plan para la iniciativa de seguridad del software enfocándose primero en las metas y objetivos, y permitir que las actividades vayan apareciendo por sí mismas. Suele ser muy útil, en estos casos, crear un cronograma para la implementación del proyecto.

Por supuesto, aprender de la experiencia también es una buena estrategia. Con ese fin, esta sección está dedicada a describir el conjunto de las actividades que observamos en por lo menos 43 de las 67 organizaciones que estudiamos, y luego proporciona una tabla de las 112 actividades con una puntuación resumida que puede ser considerada como una ponderación aproximada.

Núcleo de actividades de BSIMM

De las 112 actividades observadas en BSIMM-V, hay doce actividades que realizan por lo menos 43 de las 67 empresas que fueron estudiadas (64 %), una identificada en cada práctica. Aunque no podemos concluir de forma directa que estas doce actividades son necesarias para todas las iniciativas de seguridad del software, sí podemos decir con confianza que habitualmente se las encuentra en los programas que tienen gran éxito. Esto sugiere que si usted está trabajando en una iniciativa propia, debe considerar estas doce actividades con especial cuidado (por no hablar de las otras 100).

	Doce actividades principales que todos realizan	
	Objetivo	**Actividad**
[SM1.4]	establecer las puertas de control del SSDL (pero no hacerlas obligatorias)	identificar la ubicación de las puertas de control (gates), reunir los artefactos necesarios
[CP1.2]	promover la privacidad	identificar obligaciones relativas a la información de identificación personal (IIP)
[T1.1]	promover la cultura de la seguridad en toda la organización	proveer capacitación orientada a la concientización
[AM1.2]	priorizar las aplicaciones por los datos consumidos/manipulados	crear un esquema de clasificación e inventario de datos
[SFD1.1]	crear orientación proactiva en base a las características de seguridad	construir y publicar las características de seguridad
[SR1.1]	satisfacer la demanda para las características de seguridad	crear normas de seguridad (T: características/diseño de seguridad)
[AA1.1]	comenzar con AA	llevar a cabo la revisión de las características de seguridad
[CR1.4]	impulsar la eficiencia/consistencia con automatización	utilizar herramientas automatizadas junto con la revisión manual
[ST1.3]	comenzar las pruebas de seguridad en un territorio funcional familiar	conducir pruebas con los requisitos y características de seguridad
[PT1.1]	demostrar que el código de su organización también necesita ayuda	utilizar testers de penetración externos para encontrar problemas
[SE1.2]	proporcionar una sólida base de equipos de computación/redes para el software	garantizar que se encuentra implementada la seguridad básica en los equipos de computación y redes
[CMVM1.2]	utilizar datos de las operaciones para cambiar el comportamiento del desarrollo	identificar los defectos de software encontrados en la monitorización de las operaciones y retroalimentar al desarrollo

Actividades observadas en las 67 empresas

La siguiente tabla muestra cómo muchas de las 67 empresas estudiadas adoptaron diversas actividades. Las doce actividades esenciales se destacan en amarillo. Aunque usted puede utilizar esto como una "ponderación en bruto" de las actividades según su prevalencia, la planificación de una iniciativa de seguridad del software se aborda mejor a través de metas y objetivos.

Gobernanza		Inteligencia		Puntos de Contacto SSDL		Despliegue	
Actividad	Observado	Actividad	Observado	Actividad	Observado	Actividad	Observado
[SM1.1]	44	[AM1.1]	21	[AA1.1]	56	[PT1.1]	62
[SM1.2]	34	[AM1.2]	43	[AA1.2]	35	[PT1.2]	51
[SM1.3]	34	[AM1.3]	30	[AA1.3]	24	[PT1.3]	43
[SM1.4]	57	[AM1.4]	12	[AA1.4]	42	[PT2.2]	24
[SM1.6]	36	[AM1.5]	42	[AA2.1]	10	[PT2.3]	27
[SM2.1]	26	[AM1.6]	16	[AA2.2]	8	[PT3.1]	13
[SM2.2]	31	[AM2.1]	7	[AA2.3]	20	[PT3.2]	8
[SM2.3]	27	[AM2.2]	11	[AA3.1]	11		
[SM2.5]	20	[AM3.1]	4	[AA3.2]	4		
[SM3.1]	16	[AM3.2]	6				
[SM3.2]	6						
[CP1.1]	43	[SFD1.1]	54	[CR1.1]	24	[SE1.1]	34
[CP1.2]	52	[SFD1.2]	53	[CR1.2]	34	[SE1.2]	61
[CP1.3]	45	[SFD2.1]	26	[CR1.4]	50	[SE2.2]	31
[CP2.1]	24	[SFD2.2]	29	[CR1.5]	23	[SE2.4]	25
[CP2.2]	28	[SFD2.3]	9	[CR1.6]	25	[SE3.2]	10
[CP2.3]	29	[SFD3.1]	13	[CR2.2]	10	[SE3.3]	9
[CP2.4]	25	[SFD3.2]	9	[CR2.5]	15		
[CP2.5]	35			[CR3.1]	18		
[CP3.1]	14			[CR3.2]	4		
[CP3.2]	11			[CR3.3]	6		
[CP3.3]	8			[CR3.4]	1		
[T1.1]	50	[SR1.1]	48	[ST1.1]	51	[CMVM1.1]	59
[T1.5]	29	[SR1.2]	43	[ST1.3]	55	[CMVM1.2]	59
[T1.6]	23	[SR1.3]	45	[ST2.1]	27	[CMVM2.1]	50
[T1.7]	33	[SR1.4]	27	[ST2.3]	13	[CMVM2.2]	44
[T2.5]	9	[SR2.1]	23	[ST2.4]	11	[CMVM2.3]	30
[T2.6]	13	[SR2.2]	19	[ST3.1]	8	[CMVM3.1]	6
[T2.7]	9	[SR2.3]	19	[ST3.2]	6	[CMVM3.2]	6
[T3.1]	4	[SR2.4]	22	[ST3.3]	5	[CMVM3.3]	2
[T3.2]	4	[SR2.5]	8	[ST3.4]	7		
[T3.3]	8	[SR3.1]	12				
[T3.4]	9						
[T3.5]	5						

Apéndice: Ajustes de BSIMM4 para BSIMM-V

Debido a que BSIMM es un modelo conducido por datos, decidimos realizar ajustes al modelo en base a los datos observados entre BSIMM4 y BSIMM-V.

Con el fin de preservar la compatibilidad hacia atrás, todos los cambios se hicieron mediante la adición al modelo de nuevas etiquetas de actividad, incluso cuando una actividad se haya simplemente movido entre niveles.

Realizamos todos los cambios considerando los valores atípicos en el propio modelo y en los niveles que asignamos a las diversas actividades en las doce prácticas. Utilizamos los resultados de un análisis de desviación estándar intranivel para determinar qué actividades "atípicas" mover entre los niveles. Para ello, nos centramos en los cambios que reducen al mínimo la desviación estándar en el número promedio de las actividades observadas en cada nivel. Nuestra regla estricta y rápida fue asegurar que, en promedio, el número de actividades observadas por nivel siguiese una progresión lógica de "común" a "rara" (como se indica en nuestra discusión de los niveles).

Aquí están los cuatro cambios que hemos hecho de acuerdo con ese paradigma:

1. [SFD2.3] se convirtió en [SFD3.3] , [SFD2.3] se quitó (promoción de nivel)
2. [SR2.1] se convirtió en [SR3.2] , [SR2.1] se quitó (promoción de nivel)
3. [ST2.3] se convirtió en [ST3.5] , [ST2.3] se quitó (promoción de nivel)
4. [CR3.1] se convirtió en [CR2.6] , [CR3.1] se quitó (degradación de nivel)

También hemos considerado cuidadosamente, pero no ajustamos, las siguientes actividades: [CP2.5] , [T3.3] , [T3.4] , [AM1.4] .

La ficha de puntaje resultante para BSIMM-V se puede ver en la página 67.


VIII
VII
VI
NNE
NE
ENE
ESE
SSE
SSW